



**Approximate Dynamic Programming Algorithms
for United States Air Force Officer Sustainment**

THESIS

MARCH 2015

Joseph C. Hoecherl, Captain, USAF
AFIT-ENS-MS-15-M-126

**DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY**

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

DISTRIBUTION STATEMENT A
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

The views expressed in this document are those of the author and do not reflect the official policy or position of the United States Air Force, the United States Department of Defense or the United States Government. This material is declared a work of the U.S. Government and is not subject to copyright protection in the United States.

AFIT-ENS-MS-15-M-126

APPROXIMATE DYNAMIC PROGRAMMING ALGORITHMS FOR
UNITED STATES AIR FORCE OFFICER SUSTAINMENT

THESIS

Presented to the Faculty
Department of Operational Sciences
Graduate School of Engineering and Management
Air Force Institute of Technology
Air University
Air Education and Training Command
in Partial Fulfillment of the Requirements for the
Degree of Master of Science in Operations Research

Joseph C. Hoecherl, BS, MS
Captain, USAF

MARCH 2015

DISTRIBUTION STATEMENT A
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

APPROXIMATE DYNAMIC PROGRAMMING ALGORITHMS FOR
UNITED STATES AIR FORCE OFFICER SUSTAINMENT
THESIS

Joseph C. Hoecherl, BS, MS
Captain, USAF

Committee Membership:

Lt Col Matthew J. Robbins, Ph.D.
Chair

Raymond R. Hill, Ph.D.
Member

Darryl K. Ahner, Ph.D.
Member

Abstract

The United States Air Force (USAF) officer sustainment system involves making accession and promotion decisions for nearly 64 thousand officers annually. We formulate a discrete time stochastic Markov decision process model to examine this military workforce planning problem. The large size of the motivating problem suggests that conventional exact dynamic programming algorithms are inappropriate. As such, we propose two approximate dynamic programming (ADP) algorithms to solve the problem. We employ a least-squares approximate policy iteration (API) algorithm with instrumental variables Bellman error minimization to determine approximate policies. In this API algorithm, we use a modified version of the Bellman equation based on the post-decision state variable. Approximating the value function using a post-decision state variable allows us to find the best policy for a given approximation using a decomposable mixed integer nonlinear programming formulation. We also propose an approximate value iteration algorithm using concave adaptive value estimation (CAVE). The CAVE algorithm identifies an improved policy for a test problem based on the current USAF officer sustainment system. The CAVE algorithm obtains a statistically significant 2.8% improvement over the currently employed USAF policy, which serves as the benchmark.

*I dedicate this research to my wife and baby girl for their unconditional love,
support, and patience throughout this process.*

Acknowledgements

I would like to thank Dr. Robbins for working with me as my advisor and for his seemingly infinite patience when guiding the concepts, notation, and development of this manuscript. Great appreciation also goes to Dr. Ahner for his assistance with nuances of the CAVE algorithm. Additionally, I would like to thank Dr. Hill for his guidance in the early phases of this research and help in the development of the structure and composition of this manuscript. Special thanks goes to CPT Aaron Rettke for his invaluable assistance with the parallel computing toolbox in MATLAB; his help really did allow me do more with less. Lastly, I would like to thank Dr. Diaz, Lt Col Dougherty, and the HAF/A1 Directorate for their encouragement, data, and feedback on this line of research.

Joseph C. Hoecherl

Table of Contents

	Page
Abstract	iv
Dedication	v
Acknowledgements	vi
List of Figures	ix
List of Tables	x
I. Introduction	1
II. Literature Review	6
2.1 Manpower Planning Models	6
Markov Chain Models	6
Simulation Models	7
Network Flow Models	8
System Dynamics Models	9
Optimization Models	9
Approximate Dynamic Programming	12
2.2 Approximate Dynamic Programming Techniques	12
Approximate Policy Iteration (API)	15
Approximate Value Iteration (AVI)	17
III. Methodology & Problem Formulation	19
3.1 MDP Formulation	19
3.2 Approximate Dynamic Programming Algorithms	24
API: Least Squares Temporal Differences	24
AVI: Concave Adaptive Value Estimation	27
IV. Results	31
4.1 Defining Model Inputs and Measures	31
4.2 Small Problem Instance Definition & Results	32
4.3 Medium Problem Instance Definition & Results	34
4.4 LAF Full Problem Definition & Results	36
4.5 Computational Performance	38
V. Conclusions	40
5.1 Conclusions	40
5.2 Future Work	41

	Page
VI. Appendix	43
6.1 Appendix A	43
6.2 Appendix B	44
Bibliography	45

List of Figures

Figure		Page
1	CAVE Piecewise Linear Value Approximation	27
2	CAVE Gradient Sampling	28
3	CAVE Gradient Update	29
4	Computation Load for Simulation Years	39

List of Tables

Table		Page
1	LSTD Optimality Gap (Small Problem Instance)	33
2	ADP Optimality Gap (Small Problem Instance)	33
3	LSTD Percentage Improvement from Benchmark (Medium Problem Instance)	34
4	ADP Percentage Improvement from Benchmark (Medium Problem Instance)	35
5	ADP Percentage Improvement from Benchmark (Large Problem Instance)	37
6	Computation Times (secs)	38
7	Variables Defined	43

APPROXIMATE DYNAMIC PROGRAMMING ALGORITHMS FOR UNITED STATES AIR FORCE OFFICER SUSTAINMENT

I. Introduction

“The basic manpower problem is the following: Determine the number of personnel and their skills that best meets the future operational requirements of an enterprise.” (Gass, 1991)

The United States Air Force (USAF) is comprised of approximately 330,000 personnel who enhance national security by providing the distinctive capabilities of air and space superiority, global attack, rapid global mobility, precision engagement, information superiority, and agile combat support to the Department of Defense (DoD). The USAF, like the other branches of the military, is comprised of commissioned officers as well as the enlisted force. These two groups exhibit significantly different behaviors in regards to retention, promotion, and cross-flow between career fields. This research investigates and attempts to discover improved policies regarding management of the commissioned officer corps.

The USAF must recruit, train, and develop its personnel using limited resources. Over the last several years, the draw down from Operations Enduring Freedom and Iraqi Freedom as well as shifting domestic priorities have resulted in significant cuts to current and future outlays for acquisitions, operations, and personnel budgets. Difficult fiscal conditions emphasize the importance of having the correct mix of personnel to field a ready force. The USAF must balance the needs for officers of varying levels of experience within 90 career fields ranging from personnel officers to fighter pilots. Each of these career fields is labeled with an Air Force Specialty

Code (AFSC). The USAF has a known set of requirements (i.e., demand) and known Congressionally-mandated force size constraints.

Grades range from O-1 to O-10, with the grades O-7, O-8, O-9, and O-10 corresponding to the ranks of General Officers. The model proposed in this thesis only considers grades from O-1 to O-6, which comprise the vast majority of officers comprising the officer sustainment problem. Current USAF policy is a 100% promotion rate from O-1 to O-2 and from O-2 to O-3. This policy is primarily due to long training times and a limited performance record with which to differentiate junior officers at these grades. Moreover, officers at the grades of O-1 and O-2 frequently fill O-3 requirements. A complicating feature of the manpower planning problem faced by the USAF is the fact that senior officers are developed from junior officers only, with every recruited officer starting at the grade of O-1.

Due to these characteristics, poor personnel management decisions can have far-reaching impacts and corrective actions can take a significant amount of time to take effect. Economic factors and changes within the military environment such as operations tempo, salary, and benefits such as health care and combat pay can significantly impact retention rates (Asch *et al.*, 2008; Murray, 2004). This can result in a significant level of deviation in retention rates over time, resulting in a uncertain supply of officers to meet USAF personnel requirements. These factors can make predicting how a force structure will develop and progress a difficult process.

The current USAF personnel system determines the number of requirements for each AFSC and grade combination. Additionally, ten years of historical data are used to calculate retention rates for each combination of AFSC and number of commissioned years of service (CYOS). This information informs the development of a retention line. This retention line assumes a deterministic retention based on historical observations and no future deviation. The current model used to optimize

accessions smooths the total requirements for each AFSC over the projected retention line. Promotion decisions are made independently from the force optimization model, so the accession decisions and promotion decisions are not tied together by means of a holistic policy. By addressing only accessions policies with the static optimization model, such policies are unable to address any deviations from expected outcomes over the 30 year window. Over time, these deviations can become large, which has historically resulted in the use of significant measures to boost or lower retention such as paying bonuses to keep people in (Lakhani, 1988; Simon & Warner, 2009) or reductions in force (RIFs) to decrease the size of the force. Deviations compounded by changes in the desired force structure during times of build-up or downsizing can exacerbate the level of correction needed.

While paying bonuses has an easily calculable cost, RIFs have more subtle costs. Mone (1994) discovered that in a steady state organization, persons with low self-efficacy are significantly more likely to depart the organization than the higher performing persons with high self-efficacy. However, when an organization is actively downsizing, the correlation is reversed. The high performers begin leaving significantly more frequently than the lower performers. Additionally, Wong & McNally (1994) showed during the force reductions in the US Army in the 1990s that organizational commitment decreased significantly in the survivors of the RIFs, even when the primary means used to trim the force were voluntary separations.

We formulate a Markov decision process (MDP) model to examine the Air Force's officer sustainment problem. MDPs have several features that make them particularly suitable for this sort of workforce planning problem. An MDP can provide policies that are state-dependent, which allows for a workforce system to correct over time. State transitions can be modeled stochastically, allowing the MDP model to address the uncertainty inherent in the personnel system.

The state of the system for this problem is found by aggregating individual officers by class descriptors. The three class descriptors for the USAF officer sustainment system are career field (AFSC), commissioned years of service (CYOS), and grade (i.e., rank). The state represents the current total stock of officers, categorized by each possible combination of class descriptors. In each time period, individuals deterministically maintain their career field, stochastically transition either out of the system or to the next CYOS according to a retention parameter, and stochastically transition either to the next grade or remain in their current grade according to promotion rate decisions made within the model. The model provides a policy, π , given the current state, that indicates the number of officers to recruit for each career field (i.e., accessions) as well as the percentage of officers within specified promotion windows to be promoted. A single Line of the Air Force competitive category is examined, so promotion policies apply to officers in a specified promotion window across all career fields within the model. The contribution function imposes a cost for shortages of officers by career field and grade as well as a cost for exceeding the maximum number of allowable officers. These costs are weighted to reflect the criticality of certain AFSC and grade combinations.

The state space of the motivating problem of interest has 9,720 dimensions representing the full 54 Line of the Air Force AFSCs, 30 CYOS groups, and 6 grades. This level of dimensionality combined with the stochastic nature of the state transitions makes determining a stationary policy computationally intractable. The size of the problem suggests that development of an exact dynamic programming algorithm to obtain a solution is inappropriate.

In order to address the large size of the problem, two approximate dynamic programming (ADP) algorithms are developed to obtain non-optimal but high quality solutions. The first proposed algorithm uses least squares temporal differences with

Bellman error minimization in an approximate policy iteration framework to obtain policies. As part of the process, we simulate potential post-decision states and observe the value of a possible outcome of being in that state. After a batch of these observations is simulated, a regression is performed utilizing instrumental variables to minimize Bellman error. This algorithm uses a set of basis functions to approximate the value of the post-decision state. This approximation scheme allows the formulation of a non-linear mixed-integer program to solve the inner maximization problem, obtaining optimal actions based on the current approximation. Algorithm variants using instrumental variables regression and Latin hypercube sampling are examined.

We also use the Concave, Adaptive Value Estimation (CAVE) algorithm (Godfrey & Powell, 2002) to develop separable piecewise linear value approximations that represent the ‘cost to go’ value function for a finite-horizon formulation of the problem. This algorithm simulates potential outcomes of a given policy and uses the outcomes to update the estimate of the gradients of the value function approximation. The algorithm takes advantage of known problem structure to efficiently update the value function approximation of large numbers of policies simultaneously.

The remainder of this thesis is organized as follows. In Chapter 2, a detailed background of the USAF officer sustainment problem and relevant techniques and models is provided. Chapter 3 provides two problem formulations and the methodology used to develop and evaluate the model. Chapter 4 describes and compares the results of these models. Chapter 5 draws conclusions from these results.

II. Literature Review

This chapter examines prior manpower and workforce planning models, force management issues, Markov decision process theory, and approximate dynamic programming (ADP) theory. Prior manpower models have not typically used Markov decision processes to model large scale workforce planning problems due to the computationally intensive nature of the necessary calculations. Markov decision processes (MDP) have been used for years by those in the operations research community to address smaller discrete stochastic problems (Puterman, 1994). However, only recently has the potential of this construct begun to be realized. With the advent of approximate dynamic programming, the MDP construct can be applied to large, high-dimension problems (Powell, 2009). Application of these techniques to highly dimensional manpower and personnel problems has thus far been limited, although applications to similar resource allocation problems are well documented.

2.1 Manpower Planning Models

Markov Chain Models.

The application of Markov chain theory is common when modeling dynamic behavior in discrete time, push-flow manpower systems, where transitions are determined by fixed rates from the originating state, as opposed to pull-flow manpower systems where transitions are determined by vacancies to fill. Markov chains are typically used as descriptive models due to the lack of any mathematical programming (Wang, 2005). The result of this simplicity is that the system cannot dynamically alter these transitions internally if the result is unsatisfactory. In effect, the system models a single fixed policy.

Škulj *et al.* (2008) apply Markov chain theory to the Slovenian military manpower problem. Škulj identified and tracked transition rates between states for five years, then identified which transition rates could be adjusted. An iterative simulation was run, during which a set of transitions (actions) was identified that resulted in a close match to the desired force structure within four years. However, no static set of transition functions were found to maintain the force structure at the desired levels.

Filinkov *et al.* (2011) use Markov chains to determine how sustainable current Australian military force levels are for different possible future deployment requirements. Markov chains are used to model how a force develops over time given a current set of policies. A series of deterministic equations are used to determine the cost of a given outcome in terms of being able to sustain operational demands associated with a variety of future scenarios.

McClean (1991) compares the application of Markov chains to semi-Markov renewal models. McClean notes that the Markov chain approach is oversimplified in that it is limited to modeling a push system and requires assumptions that the probability distributions of the underlying transitions are geometric or exponential. However, renewal theory approaches are limited to modeling pull systems with constant grade sizes and are mathematically intractable when applied to reasonably sized problems.

Kinstler *et al.* (2008) develop a Markov chain model to analyze the US Navy nursing corps. Specific recruiting guidelines are identified as the cause of specified rank imbalances. Tradeoffs between levels of rank imbalance and the ability to recruit different ranks into the nursing corps are explored.

Simulation Models.

Simulation models are a valuable tool for assessing the results of different behaviors or policies within a system that may be too complex for analytical models such

as Markov chains. However, like Markov chains, simulations are primarily a descriptive tool instead of an optimizing tool. Moreover, stochastic simulations often face significant problems with auto-correlation in the output which may require advanced statistical analysis to interpret, given the variability internal to the model (Wang, 2005).

McGinnis *et al.* (1994) construct a simulation of the US Army officer professional development system to determine impacts of changes to the legal requirements for promotion due to Title IV of the Defense Reorganization Act of 1986. Their simulation is designed to evaluate potential changes to laws, policies, or internal force structure actions and inform senior personnel leaders of the consequences of these changes.

RAND uses the PILOT model to simulate how pilots are developed and trained (Mooz, 1969). The model determines the impacts of changing pilot demand levels for different aircraft systems. When the requirements for pilots of one type of aircraft increase, some pilots are transitioned from other aircraft, requiring different levels of retraining to become proficient. These transitions result in increased demand for pilots from both the aircraft being expanded as well as the aircraft supplying more experienced pilots. The PILOT model uses simulation to determine what resources are required in terms of cost, training crews, and training aircraft in order to meet potential levels of demand.

Network Flow Models.

Network flow models are another useful tool for lower dimensional problems (Gass, 1991). Network flow models have the benefit of being easy to visualize and comprehend for the non-technical decision maker. Gass (1991) shows that several manpower problems can be formulated as a general minimal-cost transshipment (flow) network.

A significant limitation of this approach is that while different flows may enter a node, these flows have only a common identifier when exiting the node. This results in a level of aggregation that may not be acceptable to all problem sets.

Mulvey (1979) compares the results of applying a network flow model, an integer program, and a simplified aggregate model to a manpower scheduling problem. The integer program is shown to require more information to implement than the network model; however, the integer program handles a wider variety of scenarios. The network flow model is easier to implement and simpler computationally.

System Dynamics Models.

System dynamics (SD) models allow the examination of continuous flows over time by incorporating feedback loops to model interactions between different structures. This technique relies heavily on the development of the appropriate equations to model the system (Wang, 2005).

Thomas *et al.* (1997) provide a system dynamics model as an alternative to a suite of tools used for US Army enlisted personnel management. A key benefit of this approach is the ability to generate causal loop diagrams to verify elemental assumptions within the model. However, this model is limited to analyzing aggregated behavior instead of differentiating by career field, grade, and time in service, which would significantly increase the level of intricacy of the required equations. The system dynamics approach for aggregated enlisted behavior is validated by replication of historical scenarios.

Optimization Models.

Optimization techniques such as linear programming (LP), integer programming (IP), goal programming (GP), and dynamic programming (DP) do provide methods

to find optimal solutions, but face significant limitations in their ability to handle certain classes of problems (Wang, 2005). Linear programming and integer programming are limited to minimizing or maximizing a single objective function and can be limited in their ability to handle stochastic problems. Dynamic programming, often formulated as a Markov Decision Process, is a powerful tool for handling sequential decision making under uncertainty, but can require significant expertise to formulate (Wang, 2005) and is computationally intensive for most real world problems (Powell, 2012). Dynamic programming has historically performed well for problems requiring a sequential allocation of resources.

Workman (2009) builds a linear program to optimize recruitment and promotion in order to develop an indigenous security force. This model replaces basic heuristics used previously with a single model that incorporates the entire enlisted and officer force across several scenarios. The model is demonstrated with current data from the Afghan National Army and provides key insights on the feasibility of potential courses of action.

The Manpower Long Range Planning System (MLRPS) (Gass *et al.*, 1988) uses Markov chains and linear programming to project and optimize the strength of the US Army over planning horizons of 10 and 20 years. This model has been used by the Army Office of the Deputy Chief of Staff of Personnel to determine which policy changes are required in order to meet force structure requirements. The 20 year horizon Manpower Planning Model differentiates the force according to grade and time in service indices and calculates the optimal long term policies to shape the total force. The 10 year horizon Manpower Requirements Model differentiates the force according to grade and skill indices and calculates the optimal policies to meet career field demands over the intermediate term.

Corbett (1995) constructs the Officer Accession/Branch Detail Model (OA/BDM),

a goal programming model to optimize accessions for US Army's Officer Personnel Management Directorate. Analysis of outputs from OA/BDM indicates that assumptions and the corresponding recommendations of the previous manpower planning program are optimistically biased. OA/BDM is also used to analyze potential courses of action to correct unbalanced overages of junior officers.

The Accession Supply Costing and Requirements Model (ASCAR) is a goal programming model that optimizes enlisted recruitment into the US Armed Forces across all branches of service (Collins *et al.*, 1983). ASCAR was used by the Congressional Budget Office and the Office of the Secretary of Defense to successfully predict whether the military would be able to simultaneously meet personnel quality goals and total end strength goals as it transitioned to an all-volunteer force.

Charnes *et al.* (1972) utilize a goal programming model for General Schedule civilian manpower management in the US Navy. This goal programming model optimizes recruitment based on requirements and cost and significantly improves the ability to optimize these factors in the presence of truncational effects such as retirement. Charnes *et al.* (1972) demonstrate their results with a hypothetical numerical illustration.

Dimitriou *et al.* (2013) use multivariate Markov chains to model a workforce transitioning within and between different departments or subgroups in a hierarchical personnel structure. This construct allows the user to evaluate the consequences of different training courses and preparation classes. Goal programming techniques are then used to achieve a desirable structure at minimum cost.

Grinold (1976) uses dynamic programming with embedded Markov chains to construct an optimal policy for naval aviator recruitment. Markov chains determine the future demand for personnel. To optimize the decisions to meet forecasted demands, Grinold (1976) uses a linear-quadratic optimal control problem, which is a special

form of Markov decision process. The linear-quadratic optimal control problem produces an optimal decision rule for any finite planning horizon that is a linear function of existing manpower stocks.

Approximate Dynamic Programming.

Approximate dynamic programming extends many of the benefits of dynamic programming to problems where solving the DP is computationally intractable (Powell, 2012). Approximate dynamic programming uses Monte Carlo simulation to sample possible outcomes of a given model and find an approximate solution using an estimated future cost function. Many approximate dynamic programs have been proven to converge to the true optimal solution if the number of sampling iterations is sufficiently large.

Song & Huang (2008) use a basic Successive Convex Approximation Method (SCAM) as well as a modified SCAM algorithm to approximate value functions for a multi-stage workforce problem with stochastic demand. Their algorithm creates a piecewise linear approximation of the value of hiring, firing, or transferring personnel for different departments with stochastic flows between departments and out of the system. The algorithm is shown to provide solutions with near-optimal values for problems small enough to be solved exactly to make the solution quality comparison.

2.2 Approximate Dynamic Programming Techniques

Dynamic programming has been demonstrated to optimize stochastic problems exceptionally well. The foundation of dynamic programming is the Bellman Equation, which provides the basis for determining actions, x , that maximize expected immediate contributions due to the state of the system at time t , S_t , and actions taken, $C(S_t, x)$, as well as expected future contributions (Bellman, 1955). The value

of these expected future contributions is captured by the expected value of the potential states to which the system may transition, $V(S_{t+1})|S_t$. The discount factor, γ , is used to weight the value of immediate contributions compared to contributions in the future. The Bellman equation is as follows:

$$V(S_t) = \max_x \mathbb{E}[C(S_t, x) + \gamma V(S_{t+1})|S_t], \quad (1)$$

where $V(S_t)$ is the value of the current state.

The primary drawback of dynamic programming for large scale manpower applications is the complexity inherent in computing expectations in the presence of multi-dimensional data. Problems with computational complexity due to dimensional data are commonly referred to as the “curse of dimensionality” (Bellman, 1957) or alternately the “three curses of dimensionality”. The three curses refer to problems arising from dimensionality in the state space, action space, and outcome space of a problem. Since dynamic programming relies on explicitly computing every possible combination of events and the value associated with this occurrence, even small levels of dimensionality can result in significant computational problems.

Approximate dynamic programming techniques are applied to solve many problems that are well suited to a dynamic programming approach, but are computationally intractable. Several techniques used in combination allow for near-optimal solutions to problems that cannot be fully evaluated by dynamic programming algorithms. ADP can resolve problems arising from dimensionality by utilizing a sampling technique to estimate an approximation of the value of a policy instead of finding the exact expectation of the value (Bellman & Dreyfus, 1959; Powell, 2009).

A critical difference between dynamic programming and approximate dynamic programming is the use by ADP of a forward pass with Monte Carlo simulation of possible random outcomes to determine values of states and actions. ADP provides

an approximate valuation and near-optimal solution, though many algorithms have been proven to converge to an optimal solution if the states are sampled sufficiently often (Ahner & Parson, 2014; Powell, 2007).

While Monte Carlo simulation relieves the problem of exactly solving for values of each state and action combination, many problems still experience significant combinatorial effects even from approximately solving for those values. Fortunately, by adapting the formulation of the problem to use a post-decision state, S_t^x , much of this combinatorial effect can be alleviated. The post-decision state can more compactly represent the possible outcomes without losing any information for many problems (Van Roy *et al.*, 1997). The value of the post-decision state, $V^x(S_t^x)$, is expressed as follows:

$$V^x(S_t^x) = \mathbb{E}[V(S_{t+1})|S_t^x]. \quad (2)$$

The following adaptation to the Bellman equation is made to make use of this powerful concept:

$$V_{t-1}^x(S_{t-1}^x) = \mathbb{E} \left[\max_x C(S_t, x) + \gamma V_t^x(S_t^x) | S_{t-1}^x \right]. \quad (3)$$

For highly dimensional problems, assigning values to every state is not computationally feasible, even with the advantages described above. To circumvent this limitation, value function approximations are used to compactly represent values of a large number of states. Approximate dynamic programming algorithms use these functions to take advantage of structure within the problem and efficiently learn about a large number of state spaces without visiting each possible outcome. Using a value function approximation in combination with approximate policy iteration is an effective method to learn about the value of different states while periodically using that information to improve the optimal policy (Powell, 2012).

Approximate Policy Iteration (API).

Approximate policy iteration fixes a specified policy and evaluates potential states and outcomes associated with that policy. This information is then used to update the policy and reevaluate. API using parametric modeling with linear basis functions has received a great deal of attention for its ability to use linear regression to derive information from a series of observations. These basis functions, also known as independent variables, covariates, or features, must be carefully selected for the algorithm to effectively approximate the value function (Bertsekas & Tsitsiklis, 1995). The selected set of basis functions is annotated \mathcal{F} , with individual features $f \in \mathcal{F}$. The crux of parametric modeling is to project the true value function onto the space of the basis functions. If the basis functions are not appropriate to the problem and the true function does not lie within the span of the basis functions, the approximation cannot converge to the true value function (Scott *et al.*, 2014). Use of these basis functions results in the following modification to the value of the post-decision state in the Bellman equation:

$$\bar{V}_t^x(S_t^x) = \sum_{f \in \mathcal{F}} \theta_f \phi_f(S_t^x) = \theta^T \phi(S_t^x), \quad (4)$$

where θ is the vector of weights, $(\theta_f)_{f \in \mathcal{F}}$, associated with the basis functions. The vector of basis function values for a post-decision state S_t^x are then defined by $\phi_f(S_t^x)$.

When using the parametric modeling approach, the selection of the optimal decision is adapted. For a given weight vector θ , the policy $X^\pi(S_t|\theta)$ is given by:

$$X^\pi(S_t|\theta) = \arg \max_x [C(S_t, x) + \gamma \theta^T \phi(S_t^x)]. \quad (5)$$

Equation 2.5 is referred to as the inner maximization problem. This problem is solved by a linear or non-linear program, depending on whether any non-linear

features have been selected as a part of $\phi(S_t^x)$. In the case of a cost function as opposed to the contribution function outlined above, the problem is treated as a minimization problem and the equation is modified as follows:

$$X^\pi(S_t|\theta) = \arg \min_x [C(S_t, x) + \gamma \theta^T \phi(S_t^x)]. \quad (6)$$

Least squares temporal differences with Bellman error minimization is an extension of parametric modeling for infinite horizon problems in which the contributions of a number of potential states are evaluated while a policy is fixed according to an initial estimate of a set of parameters, θ (Bradtke & Barto, 1996). Lagoudakis & Parr (2003) extends this algorithm to use a linear architecture to approximate state-action pairs in high dimension problems. Bellman error captures the difference between the value function approximation and the observed values being approximated. This method evaluates the value of being in a state as the observed contribution summed with the discounted value of the resulting post-decision state based on the parameter θ (Scott *et al.*, 2014).

The matrix Φ_{t-1} (see Equation 7) records the numerical value of the basis functions for N sampled post-decision states, while the matrix Φ_t records the resulting post-decision state after information has been received and a policy has been implemented. Additionally, the observed contributions are recorded in a vector C_t :

$$\Phi_{t-1} = \begin{bmatrix} \phi(S_{t-1,1}^x)^T \\ \vdots \\ \phi(S_{t-1,N}^x)^T \end{bmatrix}, \Phi_t = \begin{bmatrix} \phi(S_{t,1}^x)^T \\ \vdots \\ \phi(S_{t,N}^x)^T \end{bmatrix}, C_t = \begin{bmatrix} C_{t,1} \\ \vdots \\ C_{t,N} \end{bmatrix}. \quad (7)$$

Ordinary least squares regression is performed to identify the impact each feature

has on the observed contributions:

$$\hat{\theta} = [(\Phi_{t-1} - \gamma\Phi_t)^T(\Phi_{t-1} - \gamma\Phi_t)]^{-1}(\Phi_{t-1} - \gamma\Phi_t)^T C_t. \quad (8)$$

Least squares approximate policy iteration with instrumental variables Bellman error minimization is an efficient technique to obtain a consistent estimate of θ without modeling the noise of the system (Bradtke & Barto, 1996; Scott *et al.*, 2014). Instrumental variables are used to reduce noise within an approximate policy iteration algorithm because they are correlated with the regressor and uncorrelated with the errors and observations (Ma & Powell, 2010). Equation 2.9 shows the modified instrumental variables regression equation as presented by Scott *et al.* (2014).

$$\hat{\theta} = [(\Phi_{t-1})^T(\Phi_{t-1} - \gamma\Phi_t)]^{-1}(\Phi_{t-1})^T C_t. \quad (9)$$

Approximate Value Iteration (AVI).

An alternative to the API with a parametric modeling approach is the use of approximate value iteration. As opposed to API, where policies are fixed and then evaluated, AVI continually refines the value approximation and immediately updates the current policy accordingly. Use of separable piecewise linear approximations within an AVI framework has garnered significant attention due to their ability to map a wide variety of value functions.

The Separable Projective Approximation Routine (SPAR) algorithm (Powell, 2007) is an effective piecewise linear approximation method. SPAR samples values of states and uses a monotone structure to update the value function approximation for many states at once. Any slope approximations that violate monotonicity are averaged with the updated value to maintain structure. A key benefit to SPAR is its proof of convergence, as demonstrated by Powell (2007).

The Stochastic Hybrid Approximation Procedure (SHAPE) algorithm differs from SPAR in that it uses a nonlinear approximation of the value function (Cheung & Powell, 2000). This nonlinear approximation is updated by iteratively sampling the stochastic system to update the gradients of the value function. For this algorithm to be effective, a close initial approximation is critical. While not a separable piecewise linear approximation, SHAPE is closely related to SPAR and other piecewise linear value function approximation algorithms in that the algorithm uses stochastic gradient sampling and takes advantage of a known problem structure to develop a value approximation.

The Concave, Adaptive Value Estimation (CAVE) algorithm is very similar to the SPAR algorithm, but uses a different technique to correct monotonicity violations and can be applied to monotonicity of slopes for concave value functions (Godfrey & Powell, 2002). Instead of using averages to correct monotonicity or concavity, the CAVE algorithm expands the area being updated by the new information. Though the general CAVE algorithm does not have a convergence proof, variations of the CAVE algorithm have been proven to converge (Topaloglu & Powell, 2003; Ahner & Parson, 2014), and CAVE has been shown to outperform SPAR for many problems (Godfrey & Powell, 2001). CAVE has been applied with significant success to a number of high dimension weapon target allocation and resource allocation problems with stochastic demand.

III. Methodology & Problem Formulation

3.1 MDP Formulation

The purpose of this formulation is to provide a framework for an executable model to find the best possible policy. This policy is found by minimizing the cost of the defined objective function. This objective function is comprised of the cost of the current state as well as the expected future cost of states resulting from a combination of the current state and chosen actions. The actions selected are the accessions and promotion decisions for the following year, so the cost associated with the combination of the current state and action, typically defined as $C(S, x)$, is not affected in the present by the action x . Thus, the current contribution is defined by $C(S)$, and the consequence of a given decision is captured by the expected value of future states. The objective for the infinite horizon formulation is to minimize expected total discounted cost, with the objective function defined as:

$$\min_{\pi \in \Pi} \left(\mathbb{E}^{\pi} \left[\sum_{t=0}^{\infty} \gamma^t C(S_t) \right] \right), \quad (10)$$

where γ is the discount factor, π is a single policy, and Π is the set of all possible policies.

The alternative finite horizon objective function is defined as:

$$\min_{\pi \in \Pi} \left(\mathbb{E}^{\pi} \left[\sum_{t=0}^T C(S_t) \right] \right), \quad (11)$$

The USAF officer sustainment problem is constructed such that a single set of requirements and transition rates is assumed valid for any future time period. The parametric modeling formulation of the system is constructed as an infinite time horizon problem with annual increments. This avoids calculating a separate value for

an identical state in a different time period. The set of decision epochs is denoted as:

$$\mathcal{T} = \{1, 2, \dots\}. \quad (12)$$

For the separable piecewise linear formulation, a distinct value approximation for each decision must be developed for each time period, so the problem is constructed as a finite horizon problem with T annual increments:

$$\mathcal{T} = \{1, 2, \dots, T\}. \quad (13)$$

The state of each officer in the system is defined by an attribute vector a , composed of three numerical attributes. These attributes are numerical indices representing AFSC, grade, and CYOS. Due to the grade structure described in Chapter 1, this model consolidates the three initial grades into one index of the grade class descriptor, leaving four grades modeled. Additionally, the AFSCs are limited to the Line of the Air Force competitive category. This group consists of 54 AFSCs and contains approximately 80% of the officers in the USAF. The excluded AFSCs include medical, dental, legal, and chaplain career fields, whose behavior and constraints are sufficiently different to warrant a separate model.

Let the attribute vector a denote a specified combination of these three attributes, denoted as a_1 , a_2 , and a_3 :

$$a = \begin{pmatrix} a_1 \\ a_2 \\ a_3 \end{pmatrix} = \begin{pmatrix} \text{AFSC} \\ \text{Grade} \\ \text{CYOS} \end{pmatrix}. \quad (14)$$

The individual sets containing all iterations of a single attribute h are annotated as:

$$\mathcal{A}_h = \text{Set of all possible officer attributes } a_h. \quad (15)$$

The full set, \mathcal{A} , for the general problem contains the full range of possible combinations of m AFSCs, n grades, and q CYOSs. For the full problem examined, these values are 54, 4, and 30, respectively.

$$\mathcal{A} = \text{Set of all possible officer attribute vectors } a, \text{ where } |\mathcal{A}| = mnq. \quad (16)$$

The state of the USAF personnel system is defined by the number of resources (officers) of each attribute vector, a . Denote $S_{a,t}$ as the number of officers possessing attributes as defined by the attribute vector a .

$$S_{a,t} = \text{number of resources at time } t \text{ for attribute vector } a. \quad (17)$$

The pre-decision state is a vector of size $|\mathcal{A}|$, which is defined as $S_t = (S_{a,t})a \in \mathcal{A}$, expressed in vector form as:

$$S_t = [S_{1,1,1}, S_{1,1,2}, \dots, S_{1,1,q}, S_{1,2,1}, \dots, S_{1,n,q}, S_{2,1,1}, \dots, S_{m,n,q}], \quad (18)$$

where the subscript t is suppressed for each S_{a_1,a_2,a_3} for notational simplicity.

There are $m + n$ decisions made annually, defined by the set \mathcal{D} . Decisions $\{1, 2, \dots, m\}$ are the accessions decisions for each AFSC, determining how many people to recruit as junior officers. Decisions $\{m + 1, m + 2, \dots, m + n - 1\}$ are decisions on the ratio of eligible officers to promote. For the full problem, $m = 54$ and $n = 4$.

The action selected is defined by:

$$X_t = (X_{d,t})_{d \in \mathcal{D}}. \quad (19)$$

Additionally, there are upper bounds, β_d , and lower bounds, ζ_d , for each decision, as specified by the decision maker. For the accession decisions, these bounds arise out of pipeline considerations, such as training constraints or minimum training levels to sustain facilities. For the promotion decisions, extremely high or low values can have significant secondary effects on the quality of the force.

$$\beta_d \leq X_{d,t} \leq \zeta_d \quad \forall d \in \mathcal{D}. \quad (20)$$

A deterministic transition is made from pre-decision to post-decision state, S_t^x , with each non-promotion eligible group moving to the next CYOS index or retiring. Accessions fill the first year group (CYOS) index for each AFSC and promotion policies are attached at the end, increasing the vector size by $n - 1$. The promotion policies are included in the post-decision state because this policy determines the next transition, but the information ω as to what stochastic result this policy will generate has not been received yet. The states with the CYOS index associated with the first year of a potential grade (i.e., new promotions) are set to zero, since these will be filled by the stochastic promotion transitions. The post-decision state vector is then defined by:

$$\begin{aligned} S_t^x = & [X_1, S_{1,1,1}, S_{1,1,2}, \dots, S_{1,1,q-1}, 0, S_{1,2,1}, \dots, S_{1,n,q-1}, X_2, \\ & \dots, S_{m,n,q-1}, X_{m+1}, \dots, X_{m+n}]. \end{aligned} \quad (21)$$

Again, the subscript t is suppressed for each S_{a_1,a_2,a_3} for notational simplicity.

The cost function, C , is defined by the total sum of the shortages by grade and AFSC and the overages above the maximum number of personnel allowed in the system (i.e., the end strength). To differentiate between the criticality of shortages for various AFSCs, an AFSC criticality coefficient, $(b_{a_1})_{a_1 \in \mathcal{A}_1}$ is used to scale the shortage cost for each AFSC. Requirements by AFSC (a_1) and grade (a_2) are annotated as $(R_{a_1, a_2})_{(a_1, a_2) \in \mathcal{A}_1 \times \mathcal{A}_2}$ for all combinations of a_1 and a_2 . Let F denote the maximum end strength and let e denote the end strength criticality coefficient. The end strength criticality coefficient allows weighting the relative importance of end strength and shortages using decision maker preferences. Let C_H and C_E denote the cost due to shortage and end strength, respectively. Then,

$$C_H(S) = \sum_{a_1=1}^m \sum_{a_2=1}^n b_{a_1} \left(R_{a_1, a_2} - \sum_{a_3=1}^q S_a \right)^+, \quad (22)$$

and

$$C_E(S) = e \left(\left(\sum_{a_1=1}^m \sum_{a_2=1}^n \sum_{a_3=1}^q S_a \right) - F \right)^+. \quad (23)$$

Let $C(S)$ denote the total cost associated with a given state, which is simply the sum of the two partial costs, C_H and C_E :

$$C(S) = \sum_{a_1=1}^m \sum_{a_2=1}^n b_{a_1} \left(R_{a_1, a_2} - \sum_{a_3=1}^q S_a \right)^+ + e \left(\left(\sum_{a_1=1}^m \sum_{a_2=1}^n \sum_{a_3=1}^q S_a \right) - F \right)^+. \quad (24)$$

Promotion and retention transitions are treated as discrete stochastic functions, each following a binomial distribution. The probability of any eligible individual transitioning to the next highest grade is determined by the promotion decisions X_d for $d = \{m+1, m+2, \dots, m+n-1\}$:

$$Pr(S_{a_1, a_2+1, a_3}^x = j) = \binom{S_{a_1, a_2, a_3}^x}{j} (X_d)^j (1 - X_d)^{S_{a_1, a_2, a_3}^x - j}. \quad (25)$$

Those who are not selected for promotion remain in their current grade. After all promotion transitions are complete, the probability of any individual transitioning to the next year group (CYOS) within the individual’s current grade and AFSC is defined by the retention parameter, ρ_a , that can be derived from historic data and an environmental parameter. This additional environmental parameter can be used to scale historic retention rates to reflect beliefs about changing conditions in the future, such as economic and operations tempo impacts to force retention.

$$Pr(S_{a,t+1} = j) = \binom{S_{a,t}^x}{j} (\rho_a)^j (1 - \rho_a)^{S_{a,t}^x - j}. \quad (26)$$

Finally, the promotion decisions are no longer necessary, so the last n indices of the post-decision state are dropped as the transition back to a pre-decision state is completed.

3.2 Approximate Dynamic Programming Algorithms

API: Least Squares Temporal Differences.

Once the MDP is formulated, an iterative least squares approximate policy iteration algorithm with ordinary least squares Bellman error minimization is implemented to find solutions to the implemented formulation. Algorithm 1 is an adapted version of the algorithm presented by Scott *et al.* (2014).

At each iteration, θ is smoothed according to a step size, α . Experimentation revealed a linearly decreasing stepsize significantly outperforms a static stepsize. This allowed rapid updates early in the algorithm while benefiting from a more refined estimation as the algorithm converged to a solution.

The selected basis functions (features) are the interactions between decisions taken to some power ψ and the sums of the states S_a for various combinations of attributes.

Algorithm 1 LSTD Algorithm

```
1: Initialize  $\theta$  as a vector of zeros.
2: for  $j = 1$  to  $M$  (Policy Improvement Loop)
3:   Update  $\alpha = (M - j + 1)/(5M)$ 
4:   for  $i = 1$  to  $N$  (Policy Evaluation Loop)
5:     Simulate a random post-decision state  $S_{t-1,i}^x$ 
6:     Simulate the transition to the pre-decision state  $S_{t,i}$ 
7:     Solve MINLP for optimal decision  $X^\pi(S_t|\theta) = \arg \min_x [C(S_t) + \gamma\theta^T \phi(S_t^x)]$ 
8:     Record  $C(S_{t,i})$ ,  $\phi(S_{t-1,i}^x)$ , and  $\phi(S_{t,i}^x)$ 
9:   End
10:  Compute  $\hat{\theta} = [(\Phi_{t-1} - \gamma\Phi_t)^T(\Phi_{t-1} - \gamma\Phi_t)]^{-1}(\Phi_{t-1} - \gamma\Phi_t)^T C_t$ 
11:  Update  $\theta = (\alpha)\hat{\theta} + (1 - \alpha)\theta$ 
12: End
```

This selection helps the algorithm relate current states to potential actions and keeps the problem decomposable. For any given pre-decision state, the inner minimization problem becomes:

$$\min_X Z_1 X_1 + Z_2 X_1^2 + \dots + Z_\psi X_1^\psi + Z_{\psi+1} X_2 + \dots + Z_{\psi(m+n-1)} X_{m+n-1}^\psi, \quad (27)$$

subject to any pre-defined bounds on decisions, β_d and ζ_d . Each coefficient Z_g is determined by a number of current states and the parameter θ . The inner minimization problem is a large, decomposable mixed-integer nonlinear program with integer decisions ($X_{d,t}$) for all $d \leq m$ and continuous decisions $X_{d,t}$ for all $d > m$. Decomposability allows separation into m small integer nonlinear programs and $n - 1$ small continuous nonlinear programs. Each of these problems is solved easily.

As discussed in Chapter 2, instrumental variables have been demonstrated to significantly improve regression performance. Scott *et al.* (2014) present this adaptation to the LSTD algorithm with ordinary least squares regression by changing Step 9 in Algorithm 1 to reflect the alternative regression equation. The modified algorithm is presented as Algorithm 2:

Many approximate policy iteration algorithm implementations use uniform ran-

Algorithm 2 LSTD Algorithm with Instrumental Variables

```
1: Initialize  $\theta$  as a vector of zeros.
2: for  $j = 1$  to  $M$  (Policy Improvement Loop)
3:   Update  $\alpha = (M - j + 1)/(5M)$ 
4:   for  $i = 1$  to  $N$  (Policy Evaluation Loop)
5:     Simulate a random post-decision state  $S_{t-1,i}^x$ 
6:     Simulate the transition to the pre-decision state  $S_{t,i}$ 
7:     Solve MINLP for optimal decision  $X^\pi(S_t|\theta) = \arg \min_x [C(S_t) + \gamma\theta^T \phi(S_t^x)]$ 
8:     Record  $C(S_{t,i})$ ,  $\phi(S_{t-1,i}^x)$ , and  $\phi(S_{t,i}^x)$ 
9:   End
10:  Compute  $\hat{\theta} = [(\Phi_{t-1})^T(\Phi_{t-1} - \gamma\Phi_t)]^{-1}(\Phi_{t-1})^T C_t$ 
11:  Update  $\theta = (\alpha)\hat{\theta} + (1 - \alpha)\theta$ 
12: End
```

dom sampling of possible post-decision states in step 4. To improve the ability of the parametric regression to separate the effects of each basis function (i.e., feature) on the value function, Algorithm 3 uses Latin hypercube sampling (LHS) to generate an improved set of post-decision states. LHS designs help ensure uniform sampling across all possible dimensions thereby improving the ability of the regression to identify which regressors are significantly affecting the cost function (McKay *et al.*, 1979). The adapted algorithm with both instrumental variables and Latin hypercube sampling is shown in Algorithm 3.

Algorithm 3 IVLSTD Algorithm with Latin Hypercube Sampling

```
1: Initialize  $\theta$  as a vector of zeros.
2: for  $j = 1$  to  $M$  (Policy Improvement Loop)
3:   Construct an LHS design of  $N$  post-decision states,  $[S_{t-1,1}^x, S_{t-1,2}^x, \dots, S_{t-1,N}^x]$ 
4:   Update  $\alpha = (M - j + 1)/(5M)$ 
5:   for  $i = 1$  to  $N$  (Policy Evaluation Loop)
6:     Identify the pre-defined post-decision state  $S_{t-1,i}^x$ 
7:     Simulate the transition to the pre-decision state  $S_{t,i}$ 
8:     Solve MINLP for optimal decision  $X^\pi(S_t|\theta) = \arg \min_x [C(S_t) + \gamma\theta^T \phi(S_t^x)]$ 
9:     Record  $C(S_{t,i})$ ,  $\phi(S_{t-1,i}^x)$ , and  $\phi(S_{t,i}^x)$ 
10:  End
11:  Compute  $\hat{\theta} = [(\Phi_{t-1})^T(\Phi_{t-1} - \gamma\Phi_t)]^{-1}(\Phi_{t-1})^T C_t$ 
12:  Update  $\theta = (\alpha)\hat{\theta} + (1 - \alpha)\theta$ 
13: End
```

AVI: Concave Adaptive Value Estimation.

The alternative finite horizon formulation of the problem is solved using a version of the general CAVE algorithm proposed by Godfrey & Powell (2002). Godfrey & Powell use this algorithm to develop a piecewise linear value function approximation for a single state, $\hat{V}(s)$ for future costs, given the current state and action. We adapt this convention to develop a value function approximation for each decision, X_{d_t} , which is equivalent to the corresponding portion of the post-decision state, $S_{a_1,1,1}^x$, for $d_t = a_1$ as well as the promotion decisions.

This algorithm uses a series of breakpoints indexed by k_{d_t} , where $k_{d_t} \in \mathcal{K}_{d_t}$, and $\mathcal{K}_{d_t} = \{0, 1, \dots, k_{max}\}$. k_{max} represents the maximum number of allowable breakpoints. These breakpoints are annotated $(\nu^{k_{d_t}}, u^{k_{d_t}})$, where $\nu^{k_{d_t}}$ describes the slope of a linear segment projected from $u^{k_{d_t}}$.

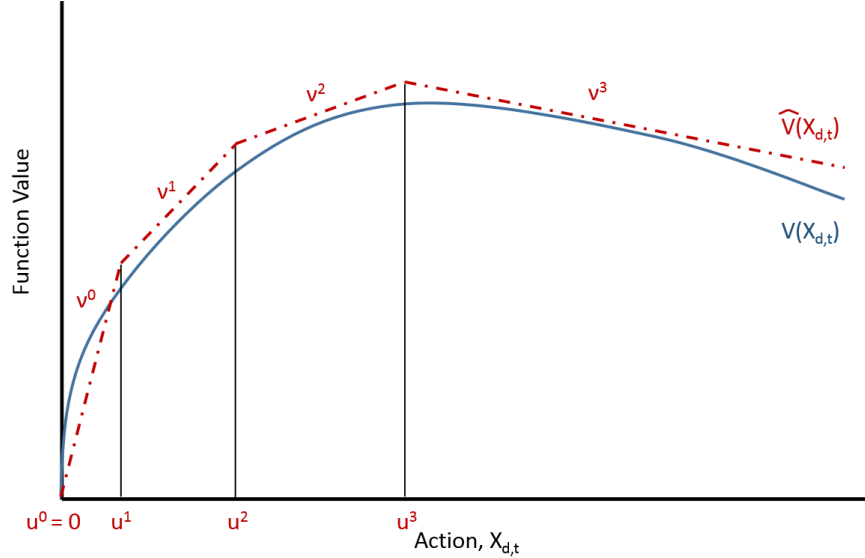


Figure 1. CAVE Piecewise Linear Value Approximation

The breakpoints $u^{k_{d_t}}$ are ordered such that $u^1 \equiv 0$ and each consecutive point is monotonically increasing:

$$u^0 < u^1 < \dots < u^{k_{max}}. \quad (28)$$

The presence of concavity in the problem structure indicates that the slopes are also monotonically decreasing:

$$\nu^0 > \nu^1 > \dots > \nu^{k_{max}}. \quad (29)$$

CAVE uses sampling of the gradients for each decision to improve its estimate of the slope for that approximation. This sampling is accomplished by a single simulation forward in time, calculating the sample gradients $\Delta_{d_t}^-(X_{d_t})$ and $\Delta_{d_t}^+(X_{d_t})$ for the segments being evaluated, $k_{d_t}^-$ and $k_{d_t}^+$.

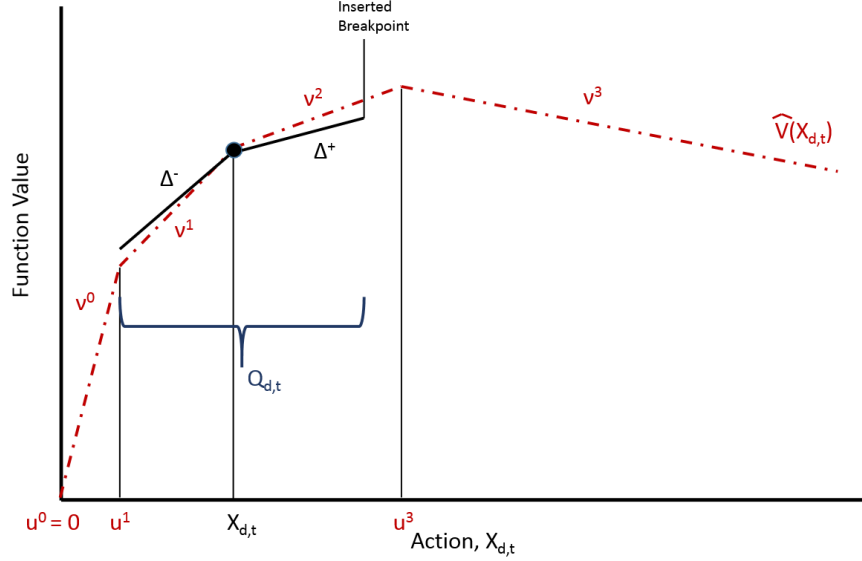


Figure 2. CAVE Gradient Sampling

A smoothing interval, Q_{d_t} for each d_t is initially set based on upper and lower interval size parameters, $\epsilon_{d_t}^-$ and $\epsilon_{d_t}^+$. This update interval is then expanded to correct any concavity violations. If necessary, new breakpoints are inserted at the ends of the update interval.

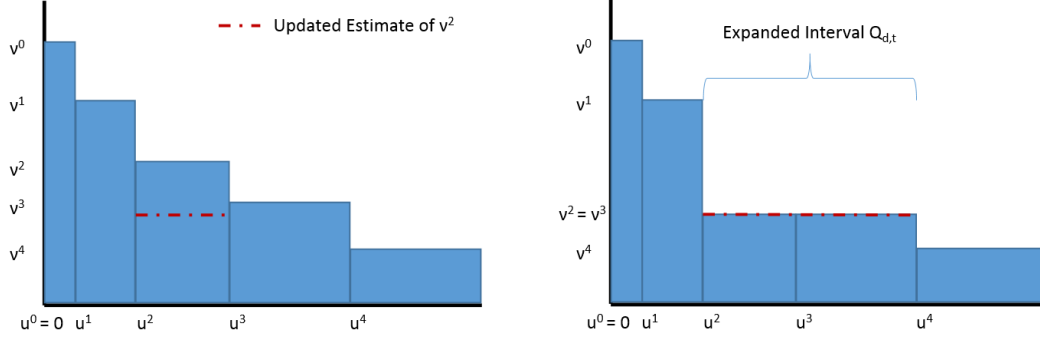


Figure 3. CAVE Gradient Update

After all of these steps are accomplished, the minimum update interval parameters, $\epsilon_{d_t}^-, \epsilon_{d_t}^+$ can be decreased to allow the algorithm to create a more granular approximation at the next time step. The step size, α can also be decreased as iterations are completed to improve value approximation convergence. The CAVE algorithm as adapted from Godfrey & Powell (2001) to the multiple decision, multiple time period problem is as follows:

Algorithm 4 CAVE Algorithm

Step 1: Initialization

- 1: For each d_t , let $\mathcal{K}_{d_t} = 0$, where $\nu_{d_t} = 0$, $u_{d_t} = 0$.
- 2: Initialize parameters $\epsilon_{d_t}^-$, $\epsilon_{d_t}^+$, and α .

3: **for** $j = 1$ **to** M
Step 2: Collect Gradient Information

- 4: For each d_t , identify the policy specified by the current value function approximation, $X_{d_t} \geq 0$.
- 5: For all decisions simultaneously, sample the gradients $\Delta_{d_t}^-(X_{d_t}, \omega)$ and $\Delta_{d_t}^+(X_{d_t}, \omega)$ over a finite time horizon with random outcome $\omega \in \Omega$

Step 3: Define Smoothing Interval

- 6: Let $k_{d_t}^- = \min\{k_{d_t} \in \mathcal{K}_{d_t} : \nu_{d_t}^{k_{d_t}} \leq (1 - \alpha)\nu_{d_t}^{k_{d_t}+1} + \alpha\Delta_{d_t}^-(X_{d_t})\}$.
- 7: Let $k_{d_t}^+ = \max\{k_{d_t} \in \mathcal{K}_{d_t} : (1 - \alpha)\nu_{d_t}^{k_{d_t}-1} + \alpha\Delta_{d_t}^+(X_{d_t}) \leq \nu_{d_t}^{k_{d_t}}\}$.
- 8: Define the smoothing interval $Q_{d_t} = \left[\min\{X_{d_t} - \epsilon_{d_t}^-, u_{d_t}^{k_{d_t}^-}\}, \max\{X_{d_t} + \epsilon_{d_t}^+, u_{d_t}^{k_{d_t}^++1}\} \right)$.
If $u_{d_t}^{(k_{d_t}^++1)}$ is undefined, then set $u_{d_t}^{(k_{d_t}^++1)} = \infty$
- 9: Create new breakpoints at X_{d_t} and the endpoints of Q_{d_t} as needed. Since a new breakpoint always divides an existing segment, the segment slopes on both sides of the new breakpoint are the same initially.

Step 4: Perform Smoothing

- 10: For each segment in the interval Q_{d_t} , update the slope according to $\nu_{d_t, new}^k = \alpha\Delta_{d_t} + (1 - \alpha)\nu_{d_t, old}^k$, where $\Delta_{d_t} = \Delta_{d_t}^-(X_{d_t})$ if $u_{d_t}^k < X_{d_t}$ and $\Delta_{d_t} = \Delta_{d_t}^+(X_{d_t})$ otherwise.
 - 11: Adjust $\epsilon_{d_t}^-$, $\epsilon_{d_t}^+$, α according to step size rules.
 - 12: **End**
-

IV. Results

4.1 Defining Model Inputs and Measures

Four performance measures provide an overview of each algorithm’s level of success for each of three problem instances that are examined. For the problem instance solved exactly with dynamic programming, results for the ADP algorithms and the benchmark policy are reported in terms of mean percentage increase in cost over the optimal policy values across all possible states. Mean relative optimality gap (MROG) is reported for reduction in cost in the small problem instance. Mean relative increase in shortages, overages, and squared deviation are also reported to allow for a more nuanced insight into the overall desirability of a given policy. A modified form of these measures are used to compare ADP policies to the benchmark (i.e., current model) policies for problems large enough to require simulation to evaluate. For these problem instances, percentages for the ADP algorithms are reported in terms of percentage improvement (i.e., decrease in cost) over the benchmark policy. We perform 50 simulations covering 50 years each, with each simulation beginning at an optimal state (i.e., no shortages or overages). Half-widths are reported at the 95% confidence level to establish statistical significance. Percent reduction in shortages (RIS), percent reduction in overages (RIO), percent reduction in cost (RIC), and percent reduction in total squared deviation (RSD) are reported. RIC, like MROG, is a direct comparison of performance regarding the objective function.

For implementations of the LSTD algorithm applied to the first two problem instances, the inner and outer loop parameters use $M = 30$ and $N = 10000$. The discount factor, γ , is set at 0.95. For each of the models, the end strength criticality coefficient, e , and AFSC criticality coefficients, $(b_{a_1})_{a_1 \in \mathcal{A}_1}$, are set equal to one. Additionally, the benchmark policy is calculated using the HAF/A1 method used to

generate the benchmark policy.

For the CAVE algorithm, T is set to twice as large as the maximum career length. This helps correctly assess the overages and shortages at the end of the career for the decisions being made. For the decisions at a time epoch to be reasonably representative of the decisions that will actually be made in the future, the impacts of those decisions are measured over a significant number of epochs. The decisions made at the end of a finite time horizon model will be biased, since the model’s reality is that only a short number of years are relevant, while the USAF has an enduring requirement for officers. Thus, T must be significantly larger than the maximum career length in order to obtain accurate and unbiased stochastic gradient samples.

4.2 Small Problem Instance Definition & Results

The small problem instance is formulated with a single AFSC ($m = 1$), single grade ($n = 1$), and four year groups ($q = 4$). Additionally, the upper accession limit is set at 6 ($\zeta_d = 6$ for $d = 1$). For the finite horizon CAVE algorithm, $T = 8$.

This small problem allows for a direct comparison of ADP and benchmark policies to the optimal policy, but fails to replicate a significant amount of the complexity inherent in larger instances of the problem. Additionally, with an effective career length of four years, random deviation is less likely to have a chance to compound and create significant shortage and overage costs for a static policy compared to a system in which the effective career length is much longer. The results bear this assertion out, with the benchmark policy beating all ADP policies tested by a significant margin.

As demonstrated in Table 1, the first order basis functions perform poorly for all algorithm variants and the higher order basis functions generally performed better. Results in each Table 1 are shaded on a color scale from green (good) to red (bad). As expected, the addition of instrumental variables and Latin hypercube sampling

generally improve algorithm performance, with the exception of the variant using third order basis functions. Of note, repeated runs did not consistently converge to similar solutions in terms of θ or solution quality, which points to potential problems with the selected basis functions. Given that good solutions consistently performed well, ten runs of each algorithm were performed, and the best performing solution was retained. This technique was utilized for the larger problem instances as well. This divergent characteristic did introduce significant variance in the performance of the LSTD algorithms throughout.

Table 1. LSTD Optimality Gap (Small Problem Instance)

Basis Functions	Random Sampling				Latin Hypercube Sampling	
	Ordinary Least Squares		Instrumental Variables		Instrumental Variables	
	MROG	MRISD	MROG	MRISD	MROG	MRISD
4th Order	32.21%	23.97%	19.60%	2.38%	18.46%	8.05%
3rd Order	26.57%	14.19%	19.21%	1.68%	25.53%	12.42%
2nd Order	33.73%	26.59%	29.51%	19.33%	28.07%	16.72%
1st Order	41.99%	50.18%	42.14%	51.05%	41.28%	49.63%

The CAVE algorithm outperforms each of the LSTD algorithm variants, as shown in Table 2. Of note is the level of performance of the non-objective function measures. Each of the algorithms generally decreased accessions, leading to increased numbers of overages and decreased numbers of shortages. The rationale for the benchmark policy is demonstrated clearly by the benchmark policy's high level of performance.

Table 2. ADP Optimality Gap (Small Problem Instance)

Algorithm		MROG	MRIS	MRIO	MRISD
LSTD	OLS	26.57%	78.38%	-90.19%	14.19%
	IV	19.21%	64.53%	-82.92%	1.68%
	IV LHS	18.46%	54.32%	-62.34%	8.05%
CAVE		15.13%	39.27%	-30.28%	8.96%
Benchmark		1.53%	-22.03%	45.84%	22.12%

4.3 Medium Problem Instance Definition & Results

A larger problem instance was examined to allow for a level of complexity that could not be obtained with a problem that could be solved to optimality while keeping the problem simple enough to easily validate the performance of the various algorithms being compared. This medium-sized problem instance was constructed with four AFSCs ($m = 4$), three grades ($n = 3$), 15 CYOSs ($q = 15$), and an upper accession limit of 50 accessions, ($\zeta_d = 50$ for $d \leq 4$). With multiple grades, promotion decisions are now assessed to determine transition rates from one grade to another. We examine 30 time epochs when implementing the CAVE algorithm ($T = 30$).

As this problem instance was examined, repetitions of all variants of the LSTD algorithm produced significantly different θ s, with significantly different solution qualities, just as observed in the small problem instance. Like the solutions from the small problem, the difference in outcomes was a function of the policy produced, as those solutions that performed well did so consistently. For each of the ten runs, the produced policy was simulated to examine solution quality, and the algorithm that performed the best in terms of the objective function was selected. The best LSTD algorithm results for each combination of sampling technique, regression technique, and basis functions are shown in Table 3.

Table 3. LSTD Percentage Improvement from Benchmark (Medium Problem Instance)

Basis Functions	Random Sampling				Latin Hypercube Sampling	
	Ordinary Least Squares		Instrumental Variables		Instrumental Variables	
	RIC	RSD	RIC	RSD	RIC	RSD
4th Order	-32.86%	-5.39%	-11.31%	-65.21%	8.29%	17.61%
3rd Order	-12.44%	-12.05%	-12.77%	5.44%	-8.97%	-46.37%
2nd Order	-12.94%	11.59%	-11.51%	4.83%	5.74%	21.88%
1st Order	-102.42%	-1025.40%	-90.53%	-867.50%	-277.51%	-4769.85%

Results that failed to improve on the benchmark results are shown in red, while results that show some level of improvement are shaded from red to green according

to the quality of the solution. As expected, the use of instrumental variables improved solution quality significantly. Latin hypercube sampling improved the solution qualities for all sets of basis functions, but improved the solution quality of complex sets of basis functions by a more significant margin than those with simpler sets. Given a constant inner loop sample size, N , algorithm variants with smaller numbers of basis functions have relatively larger amounts of information to decipher which features are impacting the observed cost. The LHS design helps algorithms with those sets too, but is at its most useful when the impact of these sample size limitations are exacerbated by large numbers of semi-correlated features. The algorithm variants with Latin hypercube sampling, instrumental variables, and either second or fourth order basis functions are the only LSTD algorithms that are able to provide policies that improve the total cost, though several other variants show improvements in squared deviation.

When observing the policies generated by the algorithms, it becomes apparent that the LSTD algorithm simplifies the problem by generating solutions that are only pseudo-dynamic. In effect, at least one of the decision policies remains static, while the other decision policies are adjusted higher or lower based on the levels of shortages or overages observed. This limitation is likely due to the value function being unable to project onto the span of the basis functions, though additional samples may have improved this problem.

Table 4. ADP Percentage Improvement from Benchmark (Medium Problem Instance)

Algorithm		RIC	Half Width	RIS	Half Width	RIO	Half Width	RSD	Half Width
LSTD	Ordinary Least Squares	-12.44%	1.60%	-23.14%	3.08%	20.93%	7.86%	-12.05%	5.76%
	Instrumental Variables	-12.77%	1.30%	-23.84%	2.48%	21.98%	6.88%	5.44%	4.98%
	IV Latin Hypercube Sampling	8.29%	1.16%	8.86%	2.19%	0.65%	9.20%	17.61%	4.20%
CAVE	Accessions	5.70%	1.05%	-5.52%	2.19%	48.96%	4.79%	35.24%	2.86%
	Accessions & Promotions	0.00%	1.81%	55.90%	1.69%	-190.47%	20.10%	-68.53%	11.66%

Two versions of CAVE were applied. The first utilizes the $m = 4$ accessions decisions and the $n - 1 = 2$ promotion decisions, while the second algorithm only

models the $m = 4$ accessions decisions. The accessions-only variant of the CAVE algorithm utilizes the benchmark policies for the $n - 1 = 2$ promotion decisions. The results using these two algorithms are shown in Table 4. Allowing the promotion rates to vary from the benchmark is a relaxation of a problem constraint. Though the relaxation of a constraint indicates that the variant with promotion decisions should be able to outperform the more constrained accessions-only model, the reverse is observed. This can be attributed to a high level of interaction between the promotion and accession policies that inhibits CAVE’s ability to converge to a quality or optimal solution. This is a significant weakness, given that non-linear interactions also exist between accession policies.

The LSTD algorithm with fourth order basis functions, instrumental variables Bellman error minimization, and Latin hypercube sampling outperformed all other algorithms tested. Additionally, this variant showed a statistically significant decrease in shortages and a statistically insignificant decrease in overages, meaning that this improvement was accomplished due to the dynamic nature of the solution without detrimentally impacting overages or shortages. The CAVE algorithm with accession policies also outperformed the benchmark policy by a statistically significant margin, with comparable performance to the second order basis function, instrumental variable, Latin hypercube variant.

4.4 LAF Full Problem Definition & Results

For the large problem instance, the most promising algorithms, LSTD with 2nd and 4th order basis functions and CAVE with accessions only, were applied to a problem of identical size to the USAF problem. This problem is formulated with 54 AFSCs ($m = 54$), four grades ($n = 4$), and 30 CYOSs ($q = 30$). For the CAVE algorithm, $T = 60$ due to the maximum 30 year career length. Three behavioral

profiles were generated to represent significant differences among observed behaviors of different AFSCs, including a high retention profile, a low retention profile, and a standard profile. These profiles represent the varying levels of demand for the skill sets of different career fields within the USAF. Each AFSC was assigned to one of these profiles, then randomly increased or decreased in size according to a uniform random distribution. Uniformly distributed unbiased variance was then introduced to the retention rates of the AFSC’s behavioral profile to generate career fields that are similar, but not identical. This procedure creates a heterogeneous mix of AFSCs with different sizes and retention rates.

Table 5. ADP Percentage Improvement from Benchmark (Large Problem Instance)

Algorithm		RIC	Half Width	RIS	Half Width	RIO	Half Width	RSD	Half Width
LSTD	2nd Order	-49.89%	0.79%	-49.80%	0.80%	-2837%	3791.96%	-41.93%	1.50%
	4th Order	-61.47%	0.87%	-30.30%	0.71%	-63228%	8651.60%	-779%	35.99%
CAVE	Accessions	2.82%	0.90%	-32.68%	2.88%	95.53%	3.21%	1.97%	1.09%

As shown in Table 5, none of the LSTD algorithms tested could improve upon the benchmark solution. In addition to the LSTD algorithm failing to generate policies that outperform the benchmark, the subjective quality of the solutions were low. Many observed policies were stationary over the simulated time, indicating that the algorithm was unable to map the value function closely enough to modify the policy dynamically, given the number of observations. This reinforces the earlier observation that the value function does not appear to project onto the span of the basis functions selected. Selection of alternate basis functions may improve this solution quality.

The CAVE algorithm demonstrates a statistically significant improvement over the benchmark policy. For this problem instance, the total overages were reduced by decreasing the number of accessions for a small number of accessions by one or two. Overages above max allowable end strength were nearly eliminated, though shortages increased significantly. Further analysis with alternative input parameters

should demonstrate potential shortage and overage trade-offs by adjusting the end strength criticality coefficient and the AFSC criticality coefficients.

4.5 Computational Performance

In order to accurately compare computational load, all computational time results are reported based on algorithm performance on a dual Intel Xeon E5-2650v2 workstation with 192 GB of RAM and MATLAB's parallel computing toolbox using 32 threads.

Table 6. Computation Times (secs)

Algorithm	Basis Functions	Small	Medium		Large	
		Solution	Solution	Simulation	Solution	Simulation
LSTD Random Sample	1st Order	57.55	242	260	-	-
LSTD Random Sample	2nd Order	62.62	271	289	-	-
LSTD Random Sample	3rd Order	64.44	294	312	-	-
LSTD Random Sample	4th Order	67.09	318	336	-	-
LSTD Latin Hypercube	1st Order	58.45	911	929	-	-
LSTD Latin Hypercube	2nd Order	66.05	924	943	118612	118772
LSTD Latin Hypercube	3rd Order	66.42	935	953	-	-
LSTD Latin Hypercube	4th Order	70.96	945	962	135799	136970
CAVE	-	-	0.58	1441	-	-
CAVE (Accessions Only)	-	0.13	0.38	953	67	167719

Table 6 shows significant fluctuations in computation time associated with the number of basis functions and sampling method utilized. Additionally, LHS improved the quality of the regression, but with a significant computational cost. This is due to the significant computational effort required to generate LHS designs for large sample sizes.

While both algorithms result in reasonable computation times, the allocation of time is significantly different. Table 6 shows that approximate policy iteration with LSTD requires a significant computational effort to develop values for the parameter, θ , but can then generate a solution for any given state very quickly. CAVE can solve for a specific state much more quickly without having to develop parameters

to describe values for all possible states. However, this process must be repeated for any given state evaluated, so simulating a large number of states eventually increases computation time to surpass that of the LSTD algorithm. Figure 4 shows this trade-off for the large problem instance.

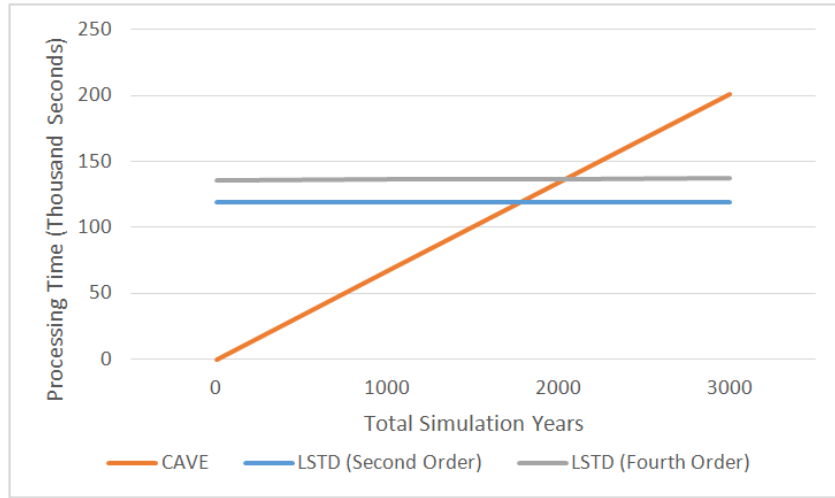


Figure 4. Computation Load for Simulation Years

V. Conclusions

5.1 Conclusions

The poor solution quality and inconsistent convergence of the LSTD algorithm indicate that the basis functions selected are not appropriate for this problem. The successes of the algorithm in improving policies for the medium problem indicate that, despite these observed limitations, these basis functions may still be able to provide improved solutions for the large problem instance with an excessive number of observations and significant additional computational resources. Additionally, these basis functions seem to be somewhat related to the true set of basis functions, given that consistent improvement in performance was observed when applying instrumental variables and space filling designs. These improvements indicate that additional information within the context of these basis functions is beneficial.

Even with an acceptable set of basis functions, there is a trade-off in sample size and sample efficiency that warrants further examination. For sample sizes that are large enough to map a problem of this size, the computational efficiency of Latin hypercube designs must be evaluated according to the level of improvement provided in calculating the regression equation and the computational demand of the design generation. This improvement and computational cost must be compared to the improvement and computational cost of additional purely random samples, given that these can improve solution quality without requiring design generation. Given the significant computational burden of generating very large designs, both in processing time and memory, case by case analysis is required in order to determine whether the addition of a Latin hypercube design is more beneficial than a computationally comparable number of additional samples.

CAVE has been shown to converge to optimal solutions for problems that do not

have some form of non-linearity due to interaction between decisions (Topaloglu & Powell, 2003). This presents as a single optimum. In this problem, each stochastic gradient sampled is only accurate given the current values of all other decisions being examined. This is a common problem for algorithms that utilize purely on-policy search to explore a decision space that has multiple local optima. CAVE is able to maximally exploit the information to find a local optima but does not have an internal mechanism to explore alternative solutions. This explains why problem instances with significant interactions, such as problems that include promotion decisions, suffer decreased levels of performance. In the USAF officer sustainment problem, these interactions exist between accessions and promotion decisions, simultaneous accession decisions for different career fields, and decisions made at different time epochs. The algorithm’s ability to improve solutions despite this non-linearity provides an idea of how much improvement may be obtained with a form of this algorithm that is modified to overcome the limitations of CAVE when applied to this type of problem structure.

5.2 Future Work

To improve the policies generated by CAVE, two immediate solutions are apparent. The first is the addition of some form of off-policy search. The use of a meta-heuristic hybrid of algorithms such as TABU search, a genetic algorithm, or GRASP to discover alternative starting locations may overcome this limitation. CAVE is then able to refine this location and converge to an optima. With sufficient sampling, this form of the algorithm is likely to enjoy a proof of convergence, given the proofs of convergence for specific instances of the CAVE algorithm and many of the heuristic search algorithms.

The second solution to the limitations of the CAVE algorithm for this type of

problem is to separate the approximation of the overage and shortage functions. The sampled stochastic gradient of the shortage function (marginal return) is then used in conjunction with the sampled stochastic gradient of the overage function (marginal cost) to evaluate which policies to adjust. This allows for the cost minimization program to correctly ascertain the impact of increasing accessions in more than one AFSC at a time. The current approach can result in many decisions changing simultaneously based on sampled information regarding overages that is no longer relevant after a single change to policy. This approach allows for a policy modification to address relatively important shortages while avoiding massive over-corrections to policies. However, this adaptation will not address nonlinearities due to the interactions of decisions made at separate time epochs.


To improve the policies generated by the approximate policy iteration algorithm, further effort in redefining these basis functions or investigating alternative regression techniques is likely to be more productive than devoting the computational resources to refine the parameter θ for the current implementation. Discovering a correct set of basis functions is historically difficult for complex stochastic problems. Additionally, a set of correct basis functions for a problem of this complexity may potentially be prohibitively large. While further trial and error may improve this algorithm, the most likely avenue for success in an approximate policy iteration framework is likely to be a non-parametric technique such as kernel regression.

VI. Appendix


6.1 Appendix A

Table 7. Variables Defined

x	Selected Action
t	Time Index
S_t	Pre-decision state at time t
γ	Discount factor
S_t^x	Post-decision state at time t
f_t	Individual basis function or feature
\mathcal{F}	Set of basis functions or features
θ	Vector of weights associated with the selected basis functions
ϕ	Vector of basis function values for a given post-decision state
N	Number of policy evaluation loops or samples
Φ	Matrix of recorded ϕ s for N sampled post-decision states
C_t	Vector of recorded costs for N sampled post-decision states
T	Maximum number of time epochs
π	A defined policy
Π	The set of all possible policies
\mathcal{T}	The set of all time epochs
a_1	AFSC or career field
a_2	Grade
a_3	CYOS or year group
a	Vector of attributes a_1 , a_2 , and a_3
h	Attribute index
\mathcal{A}_h	Set of all possible attributes a_h
\mathcal{A}	Set of all possible attributes
m	Number of AFSCs or career fields modeled
n	Number of grades modeled
q	Number of CYOS or year groups modeled
$S_{a,t}$	Number of resources at time t for attribute vector a
d	Decision index
\mathcal{D}	Set of all possible decisions
β_d	Upper bounds for decisions
ζ_d	Lower bounds for decisions
b_{a_1}	AFSC criticality coefficient
R_{a_1,a_2}	Requirements by AFSC and grade
F	Maximum allowable number of personnel in system (end strength)
e	End strength criticality coefficient
j	Transition index
ρ_a	Retention parameter
j	Policy improvement index
M	Maximum number of policy improvement loops
i	Policy evaluation index
α	Stepsize parameter
ψ	The maximum order of the polynomial used for a set of basis functions
g	Combination of decision and polynomial
Z_g	Coefficient for a decision, polynomial combination within MINLP
k_{d_t}	Breakpoint index
\mathcal{K}_{d_t}	Set of breakpoints
k_{max}	Maximum number of allowable breakpoints
$\nu^{k_{d_t}}$	Slope to the right of breakpoint k_{d_t}
$u^{k_{d_t}}$	Projection origin for breakpoint k_{d_t}
Δ_{d_t}	Observed sample gradient
Q_{d_t}	Smoothing interval
ϵ_{d_t}	Interval size parameter



Approximate Dynamic Programming Algorithms for United States Air Force Officer Sustainment



Research Objective

Generate improved policies governing accessions and promotion decisions for the USAF officer sustainment system.

Seek improvements to current policy by:

1. Modeling grade
2. Utilizing stochastic transition functions
3. Generating policies based on the current state of the personnel system

Capt Joseph C. Hoecherl

Advisor:
Lt Col Matthew J. Robbins, Ph.D.

Reader:
Raymond R. Hill, Ph.D.

Reader:
Darryl K. Anner, Ph.D.

Sponsor:
HAF/A1

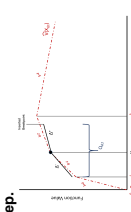
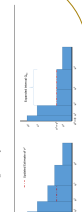
Department of Operational Sciences (ENS)

Approximate Value Iteration

The Concave Adaptive Value Estimation (CAVE) algorithm uses stochastic gradient sampling to iteratively update a separable piecewise linear value function approximation.

The Concave Adaptive Value Estimation (CAVE) algorithm uses stochastic gradient sampling to iteratively update a separable piecewise linear value function approximation.

A separate value function approximation is generated for every decision at every time step.

Approximate Dynamic Programming (ADP)

ADP algorithms are used to extend the benefits of dynamic programming to problems that are too large to solve exactly with conventional dynamic programming techniques. Two algorithmic approaches are utilized for the USAF problem:

- Approximate policy iteration (infinite horizon)
- Approximate value iteration (finite horizon)

Approximate Policy Iteration (API)

The least squares temporal differences (LSTD) algorithm uses sampled post-decision states, a mixed-order nonlinear program, and a simulation of potential outcomes of decisions to perform ordinary least squares regression to update the weighted values of a selected number of features or basis functions.

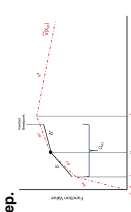
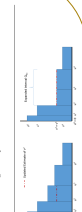
Variances of the LSTD algorithm are tested using instrumental variables regression instead of ordinary least squares and using Latin hypercube sampling instead of uniform random sampling.

Approximate Value Iteration

The Concave Adaptive Value Estimation (CAVE) algorithm uses stochastic gradient sampling to iteratively update a separable piecewise linear value function approximation.

The Concave Adaptive Value Estimation (CAVE) algorithm uses stochastic gradient sampling to iteratively update a separable piecewise linear value function approximation.

A separate value function approximation is generated for every decision at every time step.

Markov Decision Process (MDP) Formulation

MDPs are a useful tool for optimizing sequential decision making under uncertainty, typically used in dynamic programming applications.

Objective Function (Infinite, Finite):

$$\max_{\pi} \left(\mathbb{E} \left[\sum_{t=0}^{\infty} \gamma^t C(S_t) \right] \right) \quad T = \{1, 2, \dots, T\}$$

Time Space (Infinite, Finite):

$$T = \{1, 2, \dots, T\}$$

Pre Decision State Space:

$$a = \begin{pmatrix} a_1 \\ a_2 \\ a_3 \end{pmatrix} \quad \begin{matrix} \text{AFSC} \\ \text{Grade} \\ \text{CYOS} \end{matrix}$$

Action Space:

$$X_t = (X_{t,AFSC}, X_{t,Grade}, X_{t,CYOS}) \quad \forall t \in T$$

Post Decision State Space:

$$S_t' = (S_{t,AFSC}, S_{t,Grade}, S_{t,CYOS}, S_{t,AFSC-1}, S_{t,Grade-1}, S_{t,CYOS-1}, S_{t,AFSC-2}, S_{t,Grade-2}, S_{t,CYOS-2}, S_{t,AFSC-3}, S_{t,Grade-3}, S_{t,CYOS-3}, S_{t,AFSC-4}, S_{t,Grade-4}, S_{t,CYOS-4}, S_{t,AFSC-5}, S_{t,Grade-5}, S_{t,CYOS-5}, S_{t,AFSC-6}, S_{t,Grade-6}, S_{t,CYOS-6}, S_{t,AFSC-7}, S_{t,Grade-7}, S_{t,CYOS-7}, S_{t,AFSC-8}, S_{t,Grade-8}, S_{t,CYOS-8}, S_{t,AFSC-9}, S_{t,Grade-9}, S_{t,CYOS-9}, S_{t,AFSC-10}, S_{t,Grade-10}, S_{t,CYOS-10}, S_{t,AFSC-11}, S_{t,Grade-11}, S_{t,CYOS-11}, S_{t,AFSC-12}, S_{t,Grade-12}, S_{t,CYOS-12}, S_{t,AFSC-13}, S_{t,Grade-13}, S_{t,CYOS-13}, S_{t,AFSC-14}, S_{t,Grade-14}, S_{t,CYOS-14}, S_{t,AFSC-15}, S_{t,Grade-15}, S_{t,CYOS-15}, S_{t,AFSC-16}, S_{t,Grade-16}, S_{t,CYOS-16}, S_{t,AFSC-17}, S_{t,Grade-17}, S_{t,CYOS-17}, S_{t,AFSC-18}, S_{t,Grade-18}, S_{t,CYOS-18}, S_{t,AFSC-19}, S_{t,Grade-19}, S_{t,CYOS-19}, S_{t,AFSC-20}, S_{t,Grade-20}, S_{t,CYOS-20}, S_{t,AFSC-21}, S_{t,Grade-21}, S_{t,CYOS-21}, S_{t,AFSC-22}, S_{t,Grade-22}, S_{t,CYOS-22}, S_{t,AFSC-23}, S_{t,Grade-23}, S_{t,CYOS-23}, S_{t,AFSC-24}, S_{t,Grade-24}, S_{t,CYOS-24}, S_{t,AFSC-25}, S_{t,Grade-25}, S_{t,CYOS-25}, S_{t,AFSC-26}, S_{t,Grade-26}, S_{t,CYOS-26}, S_{t,AFSC-27}, S_{t,Grade-27}, S_{t,CYOS-27}, S_{t,AFSC-28}, S_{t,Grade-28}, S_{t,CYOS-28}, S_{t,AFSC-29}, S_{t,Grade-29}, S_{t,CYOS-29}, S_{t,AFSC-30}, S_{t,Grade-30}, S_{t,CYOS-30}, S_{t,AFSC-31}, S_{t,Grade-31}, S_{t,CYOS-31}, S_{t,AFSC-32}, S_{t,Grade-32}, S_{t,CYOS-32}, S_{t,AFSC-33}, S_{t,Grade-33}, S_{t,CYOS-33}, S_{t,AFSC-34}, S_{t,Grade-34}, S_{t,CYOS-34}, S_{t,AFSC-35}, S_{t,Grade-35}, S_{t,CYOS-35}, S_{t,AFSC-36}, S_{t,Grade-36}, S_{t,CYOS-36}, S_{t,AFSC-37}, S_{t,Grade-37}, S_{t,CYOS-37}, S_{t,AFSC-38}, S_{t,Grade-38}, S_{t,CYOS-38}, S_{t,AFSC-39}, S_{t,Grade-39}, S_{t,CYOS-39}, S_{t,AFSC-40}, S_{t,Grade-40}, S_{t,CYOS-40}, S_{t,AFSC-41}, S_{t,Grade-41}, S_{t,CYOS-41}, S_{t,AFSC-42}, S_{t,Grade-42}, S_{t,CYOS-42}, S_{t,AFSC-43}, S_{t,Grade-43}, S_{t,CYOS-43}, S_{t,AFSC-44}, S_{t,Grade-44}, S_{t,CYOS-44}, S_{t,AFSC-45}, S_{t,Grade-45}, S_{t,CYOS-45}, S_{t,AFSC-46}, S_{t,Grade-46}, S_{t,CYOS-46}, S_{t,AFSC-47}, S_{t,Grade-47}, S_{t,CYOS-47}, S_{t,AFSC-48}, S_{t,Grade-48}, S_{t,CYOS-48}, S_{t,AFSC-49}, S_{t,Grade-49}, S_{t,CYOS-49}, S_{t,AFSC-50}, S_{t,Grade-50}, S_{t,CYOS-50}, S_{t,AFSC-51}, S_{t,Grade-51}, S_{t,CYOS-51}, S_{t,AFSC-52}, S_{t,Grade-52}, S_{t,CYOS-52}, S_{t,AFSC-53}, S_{t,Grade-53}, S_{t,CYOS-53}, S_{t,AFSC-54}, S_{t,Grade-54}, S_{t,CYOS-54}, S_{t,AFSC-55}, S_{t,Grade-55}, S_{t,CYOS-55}, S_{t,AFSC-56}, S_{t,Grade-56}, S_{t,CYOS-56}, S_{t,AFSC-57}, S_{t,Grade-57}, S_{t,CYOS-57}, S_{t,AFSC-58}, S_{t,Grade-58}, S_{t,CYOS-58}, S_{t,AFSC-59}, S_{t,Grade-59}, S_{t,CYOS-59}, S_{t,AFSC-60}, S_{t,Grade-60}, S_{t,CYOS-60}, S_{t,AFSC-61}, S_{t,Grade-61}, S_{t,CYOS-61}, S_{t,AFSC-62}, S_{t,Grade-62}, S_{t,CYOS-62}, S_{t,AFSC-63}, S_{t,Grade-63}, S_{t,CYOS-63}, S_{t,AFSC-64}, S_{t,Grade-64}, S_{t,CYOS-64}, S_{t,AFSC-65}, S_{t,Grade-65}, S_{t,CYOS-65}, S_{t,AFSC-66}, S_{t,Grade-66}, S_{t,CYOS-66}, S_{t,AFSC-67}, S_{t,Grade-67}, S_{t,CYOS-67}, S_{t,AFSC-68}, S_{t,Grade-68}, S_{t,CYOS-68}, S_{t,AFSC-69}, S_{t,Grade-69}, S_{t,CYOS-69}, S_{t,AFSC-70}, S_{t,Grade-70}, S_{t,CYOS-70}, S_{t,AFSC-71}, S_{t,Grade-71}, S_{t,CYOS-71}, S_{t,AFSC-72}, S_{t,Grade-72}, S_{t,CYOS-72}, S_{t,AFSC-73}, S_{t,Grade-73}, S_{t,CYOS-73}, S_{t,AFSC-74}, S_{t,Grade-74}, S_{t,CYOS-74}, S_{t,AFSC-75}, S_{t,Grade-75}, S_{t,CYOS-75}, S_{t,AFSC-76}, S_{t,Grade-76}, S_{t,CYOS-76}, S_{t,AFSC-77}, S_{t,Grade-77}, S_{t,CYOS-77}, S_{t,AFSC-78}, S_{t,Grade-78}, S_{t,CYOS-78}, S_{t,AFSC-79}, S_{t,Grade-79}, S_{t,CYOS-79}, S_{t,AFSC-80}, S_{t,Grade-80}, S_{t,CYOS-80}, S_{t,AFSC-81}, S_{t,Grade-81}, S_{t,CYOS-81}, S_{t,AFSC-82}, S_{t,Grade-82}, S_{t,CYOS-82}, S_{t,AFSC-83}, S_{t,Grade-83}, S_{t,CYOS-83}, S_{t,AFSC-84}, S_{t,Grade-84}, S_{t,CYOS-84}, S_{t,AFSC-85}, S_{t,Grade-85}, S_{t,CYOS-85}, S_{t,AFSC-86}, S_{t,Grade-86}, S_{t,CYOS-86}, S_{t,AFSC-87}, S_{t,Grade-87}, S_{t,CYOS-87}, S_{t,AFSC-88}, S_{t,Grade-88}, S_{t,CYOS-88}, S_{t,AFSC-89}, S_{t,Grade-89}, S_{t,CYOS-89}, S_{t,AFSC-90}, S_{t,Grade-90}, S_{t,CYOS-90}, S_{t,AFSC-91}, S_{t,Grade-91}, S_{t,CYOS-91}, S_{t,AFSC-92}, S_{t,Grade-92}, S_{t,CYOS-92}, S_{t,AFSC-93}, S_{t,Grade-93}, S_{t,CYOS-93}, S_{t,AFSC-94}, S_{t,Grade-94}, S_{t,CYOS-94}, S_{t,AFSC-95}, S_{t,Grade-95}, S_{t,CYOS-95}, S_{t,AFSC-96}, S_{t,Grade-96}, S_{t,CYOS-96}, S_{t,AFSC-97}, S_{t,Grade-97}, S_{t,CYOS-97}, S_{t,AFSC-98}, S_{t,Grade-98}, S_{t,CYOS-98}, S_{t,AFSC-99}, S_{t,Grade-99}, S_{t,CYOS-99}, S_{t,AFSC-100}, S_{t,Grade-100}, S_{t,CYOS-100}, S_{t,AFSC-101}, S_{t,Grade-101}, S_{t,CYOS-101}, S_{t,AFSC-102}, S_{t,Grade-102}, S_{t,CYOS-102}, S_{t,AFSC-103}, S_{t,Grade-103}, S_{t,CYOS-103}, S_{t,AFSC-104}, S_{t,Grade-104}, S_{t,CYOS-104}, S_{t,AFSC-105}, S_{t,Grade-105}, S_{t,CYOS-105}, S_{t,AFSC-106}, S_{t,Grade-106}, S_{t,CYOS-106}, S_{t,AFSC-107}, S_{t,Grade-107}, S_{t,CYOS-107}, S_{t,AFSC-108}, S_{t,Grade-108}, S_{t,CYOS-108}, S_{t,AFSC-109}, S_{t,Grade-109}, S_{t,CYOS-109}, S_{t,AFSC-110}, S_{t,Grade-110}, S_{t,CYOS-110}, S_{t,AFSC-111}, S_{t,Grade-111}, S_{t,CYOS-111}, S_{t,AFSC-112}, S_{t,Grade-112}, S_{t,CYOS-112}, S_{t,AFSC-113}, S_{t,Grade-113}, S_{t,CYOS-113}, S_{t,AFSC-114}, S_{t,Grade-114}, S_{t,CYOS-114}, S_{t,AFSC-115}, S_{t,Grade-115}, S_{t,CYOS-115}, S_{t,AFSC-116}, S_{t,Grade-116}, S_{t,CYOS-116}, S_{t,AFSC-117}, S_{t,Grade-117}, S_{t,CYOS-117}, S_{t,AFSC-118}, S_{t,Grade-118}, S_{t,CYOS-118}, S_{t,AFSC-119}, S_{t,Grade-119}, S_{t,CYOS-119}, S_{t,AFSC-120}, S_{t,Grade-120}, S_{t,CYOS-120}, S_{t,AFSC-121}, S_{t,Grade-121}, S_{t,CYOS-121}, S_{t,AFSC-122}, S_{t,Grade-122}, S_{t,CYOS-122}, S_{t,AFSC-123}, S_{t,Grade-123}, S_{t,CYOS-123}, S_{t,AFSC-124}, S_{t,Grade-124}, S_{t,CYOS-124}, S_{t,AFSC-125}, S_{t,Grade-125}, S_{t,CYOS-125}, S_{t,AFSC-126}, S_{t,Grade-126}, S_{t,CYOS-126}, S_{t,AFSC-127}, S_{t,Grade-127}, S_{t,CYOS-127}, S_{t,AFSC-128}, S_{t,Grade-128}, S_{t,CYOS-128}, S_{t,AFSC-129}, S_{t,Grade-129}, S_{t,CYOS-129}, S_{t,AFSC-130}, S_{t,Grade-130}, S_{t,CYOS-130}, S_{t,AFSC-131}, S_{t,Grade-131}, S_{t,CYOS-131}, S_{t,AFSC-132}, S_{t,Grade-132}, S_{t,CYOS-132}, S_{t,AFSC-133}, S_{t,Grade-133}, S_{t,CYOS-133}, S_{t,AFSC-134}, S_{t,Grade-134}, S_{t,CYOS-134}, S_{t,AFSC-135}, S_{t,Grade-135}, S_{t,CYOS-135}, S_{t,AFSC-136}, S_{t,Grade-136}, S_{t,CYOS-136}, S_{t,AFSC-137}, S_{t,Grade-137}, S_{t,CYOS-137}, S_{t,AFSC-138}, S_{t,Grade-138}, S_{t,CYOS-138}, S_{t,AFSC-139}, S_{t,Grade-139}, S_{t,CYOS-139}, S_{t,AFSC-140}, S_{t,Grade-140}, S_{t,CYOS-140}, S_{t,AFSC-141}, S_{t,Grade-141}, S_{t,CYOS-141}, S_{t,AFSC-142}, S_{t,Grade-142}, S_{t,CYOS-142}, S_{t,AFSC-143}, S_{t,Grade-143}, S_{t,CYOS-143}, S_{t,AFSC-144}, S_{t,Grade-144}, S_{t,CYOS-144}, S_{t,AFSC-145}, S_{t,Grade-145}, S_{t,CYOS-145}, S_{t,AFSC-146}, S_{t,Grade-146}, S_{t,CYOS-146}, S_{t,AFSC-147}, S_{t,Grade-147}, S_{t,CYOS-147}, S_{t,AFSC-148}, S_{t,Grade-148}, S_{t,CYOS-148}, S_{t,AFSC-149}, S_{t,Grade-149}, S_{t,CYOS-149}, S_{t,AFSC-150}, S_{t,Grade-150}, S_{t,CYOS-150}, S_{t,AFSC-151}, S_{t,Grade-151}, S_{t,CYOS-151}, S_{t,AFSC-152}, S_{t,Grade-152}, S_{t,CYOS-152}, S_{t,AFSC-153}, S_{t,Grade-153}, S_{t,CYOS-153}, S_{t,AFSC-154}, S_{t,Grade-154}, S_{t,CYOS-154}, S_{t,AFSC-155}, S_{t,Grade-155}, S_{t,CYOS-155}, S_{t,AFSC-156}, S_{t,Grade-156}, S_{t,CYOS-156}, S_{t,AFSC-157}, S_{t,Grade-157}, S_{t,CYOS-157}, S_{t,AFSC-158}, S_{t,Grade-158}, S_{t,CYOS-158}, S_{t,AFSC-159}, S_{t,Grade-159}, S_{t,CYOS-159}, S_{t,AFSC-160}, S_{t,Grade-160}, S_{t,CYOS-160}, S_{t,AFSC-161}, S_{t,Grade-161}, S_{t,CYOS-161}, S_{t,AFSC-162}, S_{t,Grade-162}, S_{t,CYOS-162}, S_{t,AFSC-163}, S_{t,Grade-163}, S_{t,CYOS-163}, S_{t,AFSC-164}, S_{t,Grade-164}, S_{t,CYOS-164}, S_{t,AFSC-165}, S_{t,Grade-165}, S_{t,CYOS-165}, S_{t,AFSC-166}, S_{t,Grade-166}, S_{t,CYOS-166}, S_{t,AFSC-167}, S_{t,Grade-167}, S_{t,CYOS-167}, S_{t,AFSC-168}, S_{t,Grade-168}, S_{t,CYOS-168}, S_{t,AFSC-169}, S_{t,Grade-169}, S_{t,CYOS-169}, S_{t,AFSC-170}, S_{t,Grade-170}, S_{t,CYOS-170}, S_{t,AFSC-171}, S_{t,Grade-171}, S_{t,CYOS-171}, S_{t,AFSC-172}, S_{t,Grade-172}, S_{t,CYOS-172}, S_{t,AFSC-173}, S_{t,Grade-173}, S_{t,CYOS-173}, S_{t,AFSC-174}, S_{t,Grade-174}, S_{t,CYOS-174}, S_{t,AFSC-175}, S_{t,Grade-175}, S_{t,CYOS-175}, S_{t,AFSC-176}, S_{t,Grade-176}, S_{t,CYOS-176}, S_{t,AFSC-177}, S_{t,Grade-177}, S_{t,CYOS-177}, S_{t,AFSC-178}, S_{t,Grade-178}, S_{t,CYOS-178}, S_{t,AFSC-179}, S_{t,Grade-179}, S_{t,CYOS-179}, S_{t,AFSC-180}, S_{t,Grade-180}, S_{t,CYOS-180}, S_{t,AFSC-181}, S_{t,Grade-181}, S_{t,CYOS-181}, S_{t,AFSC-182}, S_{t,Grade-182}, S_{t,CYOS-182}, S_{t,AFSC-183}, S_{t,Grade-183}, S_{t,CYOS-183}, S_{t,AFSC-184}, S_{t,Grade-184}, S_{t,CYOS-184}, S_{t,AFSC-185}, S_{t,Grade-185}, S_{t,CYOS-185}, S_{t,AFSC-186}, S_{t,Grade-186}, S_{t,CYOS-186}, S_{t,AFSC-187}, S_{t,Grade-187}, S_{t,CYOS-187}, S_{t,AFSC-188}, S_{t,Grade-188}, S_{t,CYOS-188}, S_{t,AFSC-189}, S_{t,Grade-189}, S_{t,CYOS-189}, S_{t,AFSC-190}, S_{t,Grade-190}, S_{t,CYOS-190}, S_{t,AFSC-191}, S_{t,Grade-191}, S_{t,CYOS-191}, S_{t,AFSC-192}, S_{t,Grade-192}, S_{t,CYOS-192}, S_{t,AFSC-193}, S_{t,Grade-193}, S_{t,CYOS-193}, S_{t,AFSC-194}, S_{t,Grade-194}, S_{t,CYOS-194}, S_{t,AFSC-195}, S_{t,Grade-195}, S_{t,CYOS-195}, S_{t,AFSC-196}, S_{t,Grade-196}, S_{t,CYOS-196}, S_{t,AFSC-197}, S_{t,Grade-197}, S_{t,CYOS-197}, S_{t,AFSC-198}, S_{t,Grade-198}, S_{t,CYOS-198}, S_{t,AFSC-199}, S_{t,Grade-199}, S_{t,CYOS-199}, S_{t,AFSC-200}, S_{t,Grade-200}, S_{t,CYOS-200}, S_{t,AFSC-201}, S_{t,Grade-201}, S_{t,CYOS-201}, S_{t,AFSC-202}, S_{t,Grade-202}, S_{t,CYOS-202}, S_{t,AFSC-203}, S_{t,Grade-203}, S_{t,CYOS-203}, S_{t,AFSC-204}, S_{t,Grade-204}, S_{t,CYOS-204}, S_{t,AFSC-205}, S_{t,Grade-205}, S_{t,CYOS-205}, S_{t,AFSC-206}, S_{t,Grade-206}, S_{t,CYOS-206}, S_{t,AFSC-207}, S_{t,Grade-207}, S_{t,CYOS-207}, S_{t,AFSC-208}, S_{t,Grade-208}, S_{t,CYOS-208}, S_{t,AFSC-209}, S_{t,Grade-209}, S_{t,CYOS-209}, S_{t,AFSC-210}, S_{t,Grade-210}, S_{t,CYOS-210}, S_{t,AFSC-211}, S_{t,Grade-211}, S_{t,CYOS-211}, S_{t,AFSC-212}, S_{t,Grade-212}, S_{t,CYOS-212}, S_{t,AFSC-213}, S_{t,Grade-213}, S_{t,CYOS-213}, S_{t,AFSC-214}, S_{t,Grade-214}, S_{t,CYOS-214}, S_{t,AFSC-215}, S_{t,Grade-215}, S_{t,CYOS-215}, S_{t,AFSC-216}, S_{t,Grade-216}, S_{t,CYOS-216}, S_{t,AFSC-217}, S_{t,Grade-217}, S_{t,CYOS-217}, S_{t,AFSC-218}, S_{t,Grade-218}, S_{t,CYOS-218}, S_{t,AFSC-219}, S_{t,Grade-219}, S_{t,CYOS-219}, S_{t,AFSC-220}, S_{t,Grade-220}, S_{t,CYOS-220}, S_{t,AFSC-221}, S_{t,Grade-221}, S_{t,CYOS-221}, S_{t,AFSC-222}, S_{t,Grade-222}, S_{t,CYOS-222}, S_{t,AFSC-223}, S_{t,Grade-223}, S_{t,CYOS-223}, S_{t,AFSC-224}, S_{t,Grade-224}, S_{t,CYOS-224}, S_{t,AFSC-225}, S_{t,Grade-225}, S_{t,CYOS-225}, S_{t,AFSC-226}, S_{t,Grade-226}, S_{t,CYOS-226}, S_{t,AFSC-227}, S_{t,Grade-227}, S_{t,CYOS-227}, S_{t,AFSC-228}, S_{t,Grade-228}, S_{t,CYOS-228}, S_{t,AFSC-229}, S_{t,Grade-229}, S_{t,CYOS-229}, S_{t,AFSC-230}, S_{t,Grade-230}, S_{t,CYOS-230}, S_{t,AFSC-231}, S_{t,Grade-231}, S_{t,CYOS-231}, S_{t,AFSC-232}, S_{t,Grade-232}, S_{t,CYOS-232}, S_{t,AFSC-233}, S_{t,Grade-233}, S_{t,CYOS-233}, S_{t,AFSC-234}, S_{t,Grade-234}, S_{t,CYOS-234}, S_{t,AFSC-235}, S_{t,Grade-235}, S_{t,CYOS-235}, S_{t,AFSC-236}, S_{t,Grade-236}, S_{t,CYOS-236}, S_{t,AFSC-237}, S_{t,Grade-237}, S_{t,CYOS-237}, S_{t,AFSC-238}, S_{t,Grade-238}, S_{t,CYOS-238}, S_{t,AFSC-239}, S_{t,Grade-239}, S_{t,CYOS-239}, S_{t,AFSC-240}, S_{t,Grade-240}, S_{t,CYOS-240}, S_{t,AFSC-241}, S_{t,Grade-241}, S_{t,CYOS-241}, S_{t,AFSC-242}, S_{t,Grade-242}, S_{t,CYOS-242}, S_{t,AFSC-243}, S_{t,Grade-243}, S_{t,CYOS-243}, S_{t,AFSC-244}, S_{t,Grade-244}, S_{t,CYOS-244}, S_{t,AFSC-245}, S_{t,Grade-245}, S_{t,CYOS-245}, S_{t,AFSC-246}, S_{t,Grade-246}, S_{t,CYOS-246}, S_{t,AFSC-247}, S_{t,Grade-247}, S_{t,CYOS-247}, S_{t,AFSC-248}, S_{t,Grade-248}, S_{t,CYOS-248}, S_{t,AFSC-249}, S_{t,Grade-249}, S_{t,CYOS-249}, S_{t,AFSC-250}, S_{t,Grade-250}, S_{t,CYOS-250}, S_{t,AFSC-251}, S_{t,Grade-251}, S_{t,CYOS-251}, S_{t,AFSC-252}, S_{t,Grade-252}, S_{t,CYOS-252}, S_{t,AFSC-253}, S_{t,Grade-253}, S_{t,CYOS-253}, S_{t,AFSC-254}, S_{t,Grade-254}, S_{t,CYOS-254}, S_{t,AFSC-255}, S_{t,Grade-255}, S_{t,CYOS-255}, S_{t,AFSC-256}, S_{t,Grade-256}, S_{t,CYOS-256}, S_{t,AFSC-257}, S_{t,Grade-257}, S_{t,CYOS-257}, S_{t,AFSC-258}, S_{t,Grade-258}, S_{t,CYOS-258}, S_{t,AFSC-259}, S_{t,Grade-259}, S_{t,CYOS-259}, S_{t,AFSC-260}, S_{t,Grade-260}, S_{t,CYOS-260}, S_{t,AFSC-261}, S_{t,Grade-261}, S_{t,CYOS-261}, S_{t,AFSC-262}, S_{t,Grade-262}, S_{t,CYOS-262}, S_{t,AFSC-263}, S_{t,Grade-263}, S_{t,CYOS-263}, S_{t,AFSC-264}, S_{t,Grade-264}, S_{t,CYOS-264}, S_{t,AFSC-265}, S_{t,Grade-265}, S_{t,CYOS-265}, S_{t,AFSC-266}, S_{t,Grade-266}, S_{t,CYOS-266}, S_{t,AFSC-267}, S_{t,Grade-267}, S_{t,CYOS-267}, S_{t,AFSC-268}, S_{t,Grade-268}, S_{t,CYOS-268}, S_{t,AFSC-269}, S_{t,Grade-269}, S_{t,CYOS-269}, S_{t,AFSC-270}, S_{t,Grade-270}, S_{t,CYOS-270}, S_{t,AFSC-271}, S_{t,Grade-271}, S_{t,CYOS-271}, S_{t,AFSC-272}, S_{t,Grade-272}, S_{t,CYOS-272}, S_{t,AFSC-273}, S_{t,Grade-273}, S_{t,CYOS-273}, S_{t,AFSC-274}, S_{t,Grade-274}, S_{t,CYOS-274}, S_{t,AFSC-275}, S_{t,Grade-275}, S_{t,CYOS-275}, S_{t,AFSC-276}, S_{t,Grade-276}, S_{t,CYOS-276}, S_{t,AFSC-277}, S_{t,Grade-277}, S_{t,CYOS-277}, S_{t,AFSC-278}, S_{t,Grade-278}, S_{t,CYOS-278}, S_{t,AFSC-279}, S_{t,Grade-279}, S_{t,CYOS-279}, S_{t,AFSC-280}, S_{t,Grade-280}, S_{t,CYOS-280}, S_{t,AFSC-281}, S_{t,Grade-281}, S_{t,CYOS-281}, S_{t,AFSC-282}, S_{t,Grade-282}, S_{t,CYOS-282}, S_{t,AFSC-283}, S_{t,Grade-283}, S_{t,CYOS-283}, S_{t,AFSC-284}, S_{t,Grade-284}, S_{t,CYOS-284}, S_{t,AFSC-285}, S_{t,Grade-285}, S_{t,CYOS-285}, S_{t,AFSC-286}, S_{t,Grade-286}, S_{t,CYOS-286}, S_{t,AFSC-287}, S_{t,Grade-287}, S_{t,CYOS-287}, S_{t,AFSC-288}, S_{t,Grade-288}, S_{t,CYOS-288}, S_{t,AFSC-289}, S_{t,Grade-289}, S_{t,CYOS-289}, S_{t,AFSC-290}, S_{t,Grade-290}, S_{t,CYOS-290}, S_{t,AFSC-291}, S_{t,Grade-291}, S_{t,CYOS-291}, S_{t,AFSC-292}, S_{t,Grade-292}, S_{t,CYOS-292}, S_{t,AFSC-293}, S_{t,Grade-293}, S_{t,CYOS-293}, S_{t,AFSC-294}, S_{t,Grade-294}, S_{t,CYOS-294}, S_{t,AFSC-295}, S_{t,Grade-295}, S_{t,CYOS-295}, S_{t,AFSC-296}, S_{t,Grade-296}, S_{t,CYOS-296}, S_{t,AFSC-297}, S_{t,Grade-297}, S_{t,CYOS-297}, S_{t,AFSC-298}, S_{t,Grade-298}, S_{t,CYOS-298}, S_{t,AFSC-299}, S_{t,Grade-299}, S_{t,CYOS-299}, S_{t,AFSC-300}, S_{t,Grade-300}, S_{t,CYOS-300}, S_{t,AFSC-301}, S_{t,Grade-301}, S_{t,CYOS-301}, S_{t,AFSC-302}, S_{t,Grade-302}, S_{t,CYOS-302}, S_{t,AFSC-303}, S_{t,Grade-303}, S_{t,CYOS-303}, S_{t,AFSC-304}, S_{t,Grade-304}, S_{t,CYOS-304}, S_{t,AFSC-305}, S_{t,Grade-305}, S_{t,CYOS-305}, S_{t,AFSC-306}, S_{t,Grade-306}, S_{t,CYOS-306}, S_{t,AFSC-307}, S_{t,Grade-307}, S_{t,CYOS-307}, S_{t,AFSC-308}, S_{t,Grade-308}, S_{t,CYOS-308}, S_{t,AFSC-309}, S_{t,Grade-309}, S_{t,CYOS-309}, S_{t,AFSC-310}, S_{t,Grade-310}, S_{t,CYOS-310}, S_{t,AFSC-311}, S_{t,Grade-311}, S_{t,CYOS-311}, S_{t,AFSC-312}, S_{t,Grade-312}, S_{t,CYOS-312}, S_{t,AFSC-313}, S_{t,Grade-313}, S_{t,CYOS-313}, S_{t,AFSC-314}, S_{t,Grade-314}, S_{t,CYOS-314}, S_{t,AFSC-315}, S_{t,Grade-315}, S_{t,CYOS-315}, S_{t,AFSC-316}, S_{t,Grade-316}, S_{t,CYOS-316}, S_{t,AFSC-317}, S_{t,Grade-317}, S_{t,CYOS-317}, S_{t,AFSC-318}, S_{t,Grade-318}, S_{t,CYOS-318}, S_{t,AFSC-319}, S_{t,Grade-319}, S_{t,CYOS-319}, S_{t,AFSC-320}, S_{t,Grade-320}, S_{t,CYOS-320}, S_{t,AFSC-321}, S_{t,Grade-321}, S_{t,CYOS-321}, S_{t,AFSC-322}, S_{t,Grade-322}, S_{t,CYOS-322}, S_{t,AFSC-323}, S_{t,Grade-323}, S_{t,CYOS-323}, S_{t,AFSC-324}, S_{t,Grade-324}, S_{t,CYOS-324}, S_{t,AFSC-325}, S_{t,Grade-325}, S_{t,CYOS-325}, S_{t,AFSC-326}, S_{t,Grade-326}, S_{t,CYOS-326}, S_{t,AFSC-327}, S_{t,Grade-327}, S_{t,CYOS-327}, S_{t,AFSC-328}, S_{t,Grade-328}, S_{t,CYOS-328}, S_{t,AFSC-329}, S_{t,Grade-329}, S_{t,CYOS-329}, S_{t,AFSC-330}, S_{t,Grade-330}, S_{t,CYOS-330}, S_{t,AFSC-331}, S_{t,Grade-331}, S_{t,CYOS-331}, S_{t,AFSC-332}, S_{t,Grade-332}, S_{t,CYOS-332}, S_{t,AFSC-333}, S_{t,Grade-333}, S_{t,CYOS-333}, S_{t,AFSC-334}, S_{t,Grade-334}, S_{t,CYOS-334}, S_{t,AFSC-335}, S_{t,Grade-335}, S_{t,CYOS-335}, S_{t,AFSC-336}, S_{t,Grade-336}, S_{t,CYOS-336}, S_{t,AFSC-337}, S_{t,Grade-337}, S_{t,CYOS-337}, S_{t,AFSC-338}, S_{t,Grade-338}, S_{t,CYOS-338}, S_{t,AFSC-339}, S_{t,Grade-339}, S_{t,CYOS-339}, S_{t,AFSC-340}, S_{t,Grade-340}, S_{t,CYOS-340}, S_{t,AFSC-341}, S_{t,Grade-341}, S_{t,CYOS-341}, S_{t,AFSC-342}, S_{t,Grade-342}, S_{t,CYOS-342}, S_{t,AFSC-343}, S_{t,Grade-343}, S_{t,CYOS-343}, S_{t,AFSC-344}, S_{t,Grade-344}, S_{t,CYOS-344}, S_{t,AFSC-345}, S_{t,Grade-345}, S_{t,CYOS-345}, S_{t,AFSC-346}, S_{t,Grade-346}, S_{t,CYOS-346}, S_{t,AFSC-347}, S_{t,Grade-347}, S_{t,CYOS-347}, S_{t,AFSC-348}, S_{t,Grade-348}, S_{t,CYOS-348}, S_{t,AFSC-349}, S_{t,Grade-349}, S_{t,CYOS-349}, S_{t,AFSC-350}, S_{t,Grade-350}, S_{t,CYOS-350},$$

Bibliography

- Ahner, DK, & Parson, CR. 2014. Optimal multi-stage allocation of weapons to targets using adaptive dynamic programming. *Optimization Letters*, 1–13.
- Asch, BJ, Hosek, J, Mattock, M, & Panis, C. 2008. *Assessing Compensation Reform: Research in Support of the 10th Quadrennial Review of Military Compensation*. Tech. rept. DTIC Document.
- Bellman, R. 1955. Functional equations in the theory of dynamic programming. V. Positivity and quasi-linearity. *Proceedings of the National Academy of Sciences of the United States of America*, **41**(10), 743.
- Bellman, R. 1957. *Dynamic Programming*. 1 edn. Princeton, NJ, USA: Princeton University Press.
- Bellman, R, & Dreyfus, S. 1959. Functional approximations and dynamic programming. *Mathematical Tables and Other Aids to Computation*, 247–251.
- Bertsekas, DP, & Tsitsiklis, JN. 1995. Neuro-dynamic programming: an overview. *Pages 560–564 of: Proceedings of the 34th IEEE Conference on Decision and Control, 1995.*, vol. 1. IEEE.
- Bradtke, SJ, & Barto, AG. 1996. Linear least-squares algorithms for temporal difference learning. *Machine Learning*, **22**(1-3), 33–57.
- Charnes, A, Cooper, WW, & Sholtz, D. 1972. A model for civilian manpower management and planning in the US Navy. *DOCUMENT RESUME VT 020 349 Charnes, A.; And Others Studies in Manpower Planning. Carnegie-Mellon Univ., Pittsburgh, Pa. Management*.
- Cheung, RKM, & Powell, WB. 2000. SHAPE-A stochastic hybrid approximation procedure for two-stage stochastic programs. *Operations Research*, **48**(1), 73–79.
- Collins, RW, Gass, SI, & Rosendahl, EE. 1983. The ASCAR model for evaluating military manpower policy. *Interfaces*, **13**(3), 44–53.
- Corbett, JC. 1995. *Military Manpower Planning: Optimization Modeling for the Army Officer Accession/Branch Detail Program*. Tech. rept. DTIC Document.
- Dimitriou, VA, Georgiou, AC, & Tsantas, N. 2013. The multivariate non-homogeneous Markov manpower system in a departmental mobility framework. *European Journal of Operational Research*, **228**(1), 112–121.
- Filinkov, A, Richmond, M, Nicholson, R, Alshansky, M, & Stewien, J. 2011. Modelling personnel sustainability: a tool for military force structure analysis. *Journal of the Operational Research Society*, **62**(8), 1485–1497.

- Gass, SI. 1991. Military manpower planning models. *Computers & Operations Research*, **18**(1), 65–73.
- Gass, SI, Collins, RW, Meinhardt, CW, Lemon, DM, & Gillette, MD. 1988. OR PracticeThe Army Manpower Long-Range Planning System. *Operations Research*, **36**(1), 5–17.
- Godfrey, GA, & Powell, WB. 2001. An adaptive, distribution-free algorithm for the newsvendor problem with censored demands, with applications to inventory and distribution. *Management Science*, **47**(8), 1101–1112.
- Godfrey, GA, & Powell, WB. 2002. An adaptive dynamic programming algorithm for dynamic fleet management, I: Single period travel times. *Transportation Science*, **36**(1), 21–39.
- Grinold, RC. 1976. Manpower planning with uncertain requirements. *Operations Research*, **24**(3), 387–399.
- Kinstler, DP, Johnson, RW, Richter, A, & Kocher, K. 2008. Navy Nurse Corps manpower management model. *Journal of Health Organization and Management*, **22**(6), 614–626.
- Lagoudakis, MG, & Parr, R. 2003. Least-squares policy iteration. *The Journal of Machine Learning Research*, **4**, 1107–1149.
- Lakhani, H. 1988. The effect of pay and retention bonuses on quit rates in the US Army. *Industrial and Labor Relations Review*, 430–438.
- Ma, J, & Powell, WB. 2010. Convergence analysis of on-policy LSPI for multi-dimensional continuous state and action-space mdps and extension with orthogonal polynomial approximation. *Submitted to SIAM Journal of Control and Optimization*.
- McClean, S. 1991. Manpower planning models and their estimation. *European Journal of Operational Research*, **51**(2), 179–187.
- McGinnis, ML, Kays, JL, & Slaten, P. 1994. Computer simulation of US Army officer professional development. *Pages 813–820 of: Proceedings of the 26th Conference on Winter Simulation*. Society for Computer Simulation International.
- McKay, MD, Beckman, RJ, & Conover, WJ. 1979. Comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics*, **21**(2), 239–245.
- Mone, MA. 1994. Relationships between self-concepts, aspirations, emotional responses, and intent to leave a downsizing organization. *Human Resource Management*, **33**(2), 281–298.

- Mooz, WE. 1969. The Pilot Training Study: Personnel Flow and the PILOT Model.
- Mulvey, JM. 1979. Strategies in modeling: A personnel scheduling example. *Interfaces*, **9**(3), 66–77.
- Murray, CT. 2004. Military Compensation: Balancing Cash and Noncash Benefits. DTIC Document.
- Powell, WB. 2007. *Approximate Dynamic Programming: Solving the curses of dimensionality*. Vol. 703. John Wiley & Sons.
- Powell, WB. 2009. What you should know about approximate dynamic programming. *Naval Research Logistics*, **56**(3), 239–249.
- Powell, WB. 2012. Perspectives of approximate dynamic programming. *Annals of Operations Research*, 1–38.
- Puterman, ML. 1994. *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons.
- Scott, WR, Powell, WB, & Moazehi, S. 2014. Least Squares Policy Iteration with Instrumental Variables vs. Direct Policy Search: Comparison Against Optimal Benchmarks Using Energy Storage. *arXiv preprint arXiv:1401.0843*.
- Simon, CJ, & Warner, JT. 2009. The supply price of commitment: evidence from the Air Force enlistment bonus program. *Defence and Peace Economics*, **20**(4), 269–286.
- Škulj, D, Vehovar, V, & Štamfelj, D. 2008. The modelling of manpower by Markov chains—a case study of the Slovenian armed forces. *Informatica*, **32**(3), 289–91.
- Song, H, & Huang, HC. 2008. A successive convex approximation method for multistage workforce capacity planning problem with turnover. *European Journal of Operational Research*, **188**(1), 29–48.
- Thomas, DA, Kwinn, BT, McGinnis, M, Bowman, BA, & Entner, MD. 1997. The US army enlisted personnel system: a system dynamics approach. *Pages 1263–1267 of: Computational Cybernetics and Simulation., 1997 IEEE International Conference on Systems, Man, and Cybernetics, 1997.*, vol. 2. IEEE.
- Topaloglu, H, & Powell, WB. 2003. An algorithm for approximating piecewise linear concave functions from sample gradients. *Operations Research Letters*, **31**(1), 66–76.
- Van Roy, B, Bertsekas, DP, Lee, Y, & Tsitsiklis, JN. 1997. A neuro-dynamic programming approach to retailer inventory management. *Pages 4052–4057 of: Proceedings of the 36th IEEE Conference on Decision and Control, 1997.*, vol. 4. IEEE.

- Wang, J. 2005. *A Review of Operations Research Applications in Workforce Planning and Potential Modeling of Military Training*. Tech. rept. DTIC Document.
- Wong, L, & McNally, J. 1994. Downsizing the army: Some policy implications affecting the survivors. *Armed Forces & Society*, **20**(2), 199–216.
- Workman, PE. 2009. *Optimizing security force generation*. Ph.D. thesis, Monterey, California. Naval Postgraduate School.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

The public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden to Department of Defense, Washington Headquarters Services, Directorate for Information Operations and Reports (0704-0188), 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to any penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number. **PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ADDRESS.**

1. REPORT DATE (DD-MM-YYYY) 26-03-2015		2. REPORT TYPE Master's Thesis		3. DATES COVERED (From — To) Aug 2013 – Mar 2015	
4. TITLE AND SUBTITLE Approximate Dynamic Programming Algorithms for United States Air Force Officer Sustainment				5a. CONTRACT NUMBER	
				5b. GRANT NUMBER	
				5c. PROGRAM ELEMENT NUMBER	
6. AUTHOR(S) Hoecherl, Joseph C., Captain, USAF				5d. PROJECT NUMBER	
				5e. TASK NUMBER	
				5f. WORK UNIT NUMBER	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Air Force Institute of Technology Graduate School of Engineering and Management (AFIT/EN) 2950 Hobson Way WPAFB OH 45433-7765				8. PERFORMING ORGANIZATION REPORT NUMBER AFIT-ENS-MS-15-M-126	
9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES) Gerald Diaz, Ph.D Headquarters Air Force/A1 1550 W. Perimeter Rd, Rm 4710 Joint Base Andrews NAF Washington, MD 20762-5000 gerald.diaz.civ@mail.mil				10. SPONSOR/MONITOR'S ACRONYM(S) HAF/A1	
				11. SPONSOR/MONITOR'S REPORT NUMBER(S)	
12. DISTRIBUTION / AVAILABILITY STATEMENT Distribution Statement A. Approved for Public Release; distribution unlimited.					
13. SUPPLEMENTARY NOTES This material is declared a work of the U.S. Government and is not subject to copyright protection in the United States.					
14. ABSTRACT The United States Air Force (USAF) officer sustainment system involves making accession and promotion decisions for nearly 64 thousand officers annually. We formulate a discrete time stochastic Markov decision process model to examine this military workforce planning problem. The large size of the motivating problem suggests that conventional exact dynamic programming algorithms are inappropriate. As such, we propose two approximate dynamic programming (ADP) algorithms to solve the problem. We employ a least-squares approximate policy iteration (API) algorithm with instrumental variables Bellman error minimization to determine approximate policies. In this API algorithm, we use a modified version of the Bellman equation based on the post-decision state variable. Approximating the value function using a post-decision state variable allows us to find the best policy for a given approximation using a decomposable mixed integer nonlinear programming formulation. We also propose an approximate value iteration algorithm using concave adaptive value estimation (CAVE). The CAVE algorithm identifies an improved policy for a test problem based on the current USAF officer sustainment system. The CAVE algorithm obtains a statistically significant 2.8% improvement over the currently employed USAF policy, which serves as the benchmark.					
15. SUBJECT TERMS manning, officer sustainment, workforce planning, approximate dynamic programming, ADP, approximate policy iteration, approximate value iteration, least squares temporal differences, LSTD, instrumental variables, Concave Adaptive Value Estimation, CAVE					
16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Lt Col Matthew J.D. Robbins, Ph.D.
U	U	U	UU	60	19b. TELEPHONE NUMBER (include area code) (937) 255-3636, x4539;matthew.robbs@afit.edu